MASTER EN SCIENCES INFORMATIQUES

Un logiciel interactif de programmation multiobjectifs

Aguilar V., Victor

Award date:
1984

Awarding institution:
Universite de Namur

FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX - NAMUR

INSTITUT D'INFORMATIQUE

UN LOGICIEL INTERACTIF DE PROGRAMMATION MULTIOBJECTIFS

Mémoire présenté par

Victor AGUILAR V.

en vue de l'obtention du grade de

Licencié et Maître en Informatique

Promoteur: J. FICHEFET

ANNEE ACADEMIQUE 1983-1984

A: YOLY

VICO

GABY

Je remercie tout d'abord Monsieur J. Fichefet qui a accepté la direction de ce mémoire.

Je voudrais exprimer tout particulièrement ma reconnaissance à Monsieur Jean-Paul Leclercq pour l'aide efficace apportée au long de ce travail.

Je remercie également Monsieur M. Debar pour la mise au point de quelques routines de Lamps, essentielles au démarrage de la programmation de ce mémoire.

Pour terminer, je tiens à remercier tous ceux qui de près ou de loin ont contribué à la réalisation de ce projet et tout spécialement les personnes dont l'aide fut précieuse pendant la rédaction.

TABLE DE MATIERES

TABLE DE MATIERES

Chapitre 1: INTRODUCTION

La programmation linéaire multiobjectifs est une partie de la programmation mathématique qui a été développée afin de répondre à la complexité des situations réelles et plus particulièrement celles où on veut assigner des ressources limitées à un certain nombre d'activités.

Au départ, ce type de problèmes a été résolu par la programmation linéaire simple, dont le but est d'optimiser un seul objectif (par exemple, maximiser les profits ou minimiser les coûts de production).

Si on regarde de plus près une entreprise industrielle, par exemple, en plus de l'objectif évident du profit, il faut tenir compte au moins de deux autres aspects prioritaires qui sont d'une part de satisfaire les aspirations sociales et économiques des ouvriers, et d'autre part de protéger l'environnement. On se trouve ici face à une situation qui engendre trois objectifs qui devraient être optimisés simultanément. C'est à ce genre d'applications que s'adresse la programmation linéaire multiobjectifs.

Du fait même que les différents objectifs sont contradictoires entre eux, il est imposible de trouver une solution "optimale"; on se contente alors d'une solution la "plus satisfaisante" pour le décideur, qui dans notre exemple peut être le gérant de l'entreprise.

Il ne faut pas croire que la programmation linéaire multiobjectifs va résoudre le problème posé; elle permet d'aider le décideur dans le choix d'une solution parmi celles possibles.

Parmi les méthodes de programmation linéaire multiobjectifs, les interactives (procédures constituées de phases de calcul et de dialogue alternées) s'adaptent mieux à la philosophie de la programmation linéaire multiobjectifs. En effet, le dialogue avec le décideur lui permet de maîtriser les solutions possibles tout en orientant l'analyse du problème d'après ses préférences.

Le but poursuivi dans ce mémoire est la mise au point d'un logiciel de programmation linéaire multiobjectifs comprenant les méthodes interactives les plus répandues et utilisant comme outil de base le logiciel de programmation linéaire simple LAMPS, disponible sur l'ordinateur DEC-20 de l'Institut d'Informatique.

La première partie de ce travail consiste en une étude et analyse des problèmes de programmation linéaire multiobjectifs (PLMO).

Le deuxième chapitre donne la formulation mathématique des problèmes de PLMO ainsi que les définitions des notions générales, liées à de tels problèmes.

Au chapitre 3, nous présentons, une étude des différentes méthodes de résolution des PLMO. Sont analysées, de façon détaillée les métodes du Critère Global, de Geoffrion, du Goal Programming interactif, la méthode Stem et celle de Zionts et Wallenius. On y décrit également les algorithmes pour ces méthodes

Le chapitre 4 explique les caratéristiques du logiciel de programmation linéaire "LAMPS". C'est celui-ci qui a été utilisé comme programme utilitaire pour la résolution des problèmes de PLMO par les méthodes implémentées.

La deuxième partie du travail concerne l'implémentation des méthodes proprement dite. Un des objectifs principaux était de rendre l'introduction des données très facile et agréable pour l'utilisateur. Pour ce faire, des programmes de saisie de données ont été écrits. Ils permettent à l'utilisateur de fournir les données sous forme d'équations et les transforment en une forme exigée par LAMPS.

Après l'introduction ds données, celles-ci sont conservées dans un fichier (fichier de données brutes), qui peut servir directement, ou après modifications, à la résolution future.

De même, nous avons voulu rendre souple le processus de résolution du problème. Celui-ci se présente de façon itérative; après chaque exécution, on peut soit changer de méthode de résolution, soit actualiser les données. Ainsi, le logiciel permet de mettre à jour les coefficients de la matrice du problème tandis que si l'utilisateur veut ajouter/supprimer une équation ou une variable au problème, il doit modifier le fichier des données brutes.

Pour terminer, le chapitre 6 présente le manuel de l'utilisateur et un exemple de résolution d'un problème de nutrition formulé dans [14].

Chapitre 2: TERMINOLOGIE, DEFINITIONS ET NOTATIONS

2.1. Formulation d'un problème de programmation multiobjectifs.

Mathématiquement, un problème de programmation multiobjectifs peut être représenté de la façon suivante :

$$\max [\ z1(\underline{x}),.....,zk(\underline{x})\ ] \qquad (2.1)$$

sous les contraintes (s.c.) :

$$gi(x) \leq 0;\ i = 1,m \qquad (2.2)$$

$$xj \geq 0;\ j = 1,n \qquad (2.3)$$

où:

- $\underline{x} = (x1,x2,.....,xn) \in \mathbb{R}^n$, sont les actions ou objets (encore appelées variables de décision) auxquels on veut assigner des ressources limitées.
- $zi(\underline{x})$ sont les fonctions représentant les objectifs (k au total) à maximiser[1] . Une fonction objectif s'écrit, dans le cas linéaire:

$$zi(x) = \sum_{j=1}^{n} cij.xj \qquad (2.4)$$

où les cij sont les profits ou coûts attachés à chaque action ou objet j dans l'objectif i.

- $gi(\underline{x})$ sont les équations ou inégalités (m au total) représentant les contraintes sur la disponibilité initiale des ressources.

Les fonctions $zi(\underline{x})$ et $gi(\underline{x})$ peuvent être linéaires ou non linéaires.

Dans le présent travail, on s'occupera du cas où les 2 types de fonctions sont linéaires, c'est-à-dire de la Programmation linéaire Multiobjectifs (PLMO).

---

(1) Si un des objectifs est à minimiser, il suffit de changer de signe, pour le ramener à une maximisation.

2.2. Solution possible.

Une solution possible ou réalisable est une solution qui satisfait (2.2) et (2.3).

L'ensemble de solutions possibles est connu comme "région possible" et défini de la façon suivante :

$$X = \{\underline{x} \in \mathbb{R}^n \mid gi(\underline{x}) \leq 0,\ xj \geq 0,\ \forall i,j\} \qquad (2.5)$$

De même, on peut définir l'ensemble des valeurs possibles des objectifs comme :

$$B = \{\underline{z}(\underline{x}) \mid \underline{x} \in X\} \qquad (2.6)$$

chaque vecteur $\underline{x}$ a son élément correspondant $[z1(\underline{x}), \ldots zk(\underline{x})]$ dans l'ensemble d'objectifs possibles.

Si on note $\underline{z}(\underline{x})$, le vecteur des fonctions objectifs, les formules (2.1), (2.2) et (2.3) peuvent être écrites, sous forme vectorielle :

$$\max \underline{z}(\underline{x})$$

$$\underline{x} \in X \qquad (2.7)$$

2.3. Solution optimale

Une solution optimale ou idéale au problème de programmation multiobjectif est celle qui maximise simultanément les k objectifs; on dira donc que $\hat{\underline{x}}$ est une solution optimale si et seulement si

$$\hat{\underline{x}} \in X$$

$$\underline{z}(\hat{\underline{x}}) \geq \underline{z}(\underline{x}),\ \forall \underline{x} \in X \qquad (2.8)$$

Le vecteur des objectifs pour cette solution optimale sera

$$\hat{\underline{z}}(\underline{x}) = [z1(\hat{\underline{x}}), \ldots\ldots, zk(\hat{\underline{x}})]$$

Puisque les problèmes de PLMO comprennent l'optimisation de plusieurs objectifs de nature contradictoire, on n'obtient presque jamais en général une solution optimale; on essaye plutôt alors de trouver une solution efficace.

2.4. Solution efficace

Connue aussi comme solution non-dominée ou Pareto optimale, c'est une solution dont aucun objectif ne peut être amélioré, sans le détriment simultané d'au moins un autre objectif; mathématiquement, on a une solution efficace $x^* \in X$ si

$$\begin{aligned} &\nexists \text{ un autre } \underline{x} \in X \text{ tel que :} \\ &z_i(\underline{x}) \geq z_i(\underline{x}^*) \ \forall i \\ &\text{et } z_i(\underline{x}) > z_i(\underline{x}^*) \text{ pour au moins un } i \end{aligned} \tag{2.9}$$

On définit alors l'ensemble de solutions efficaces comme :

$$S = \{\underline{x}^* \in X\} \tag{2.10}$$

De même, on peut définir l'ensemble des objectifs efficaces comme :

$$Z = \{\underline{z}(x^*) \in \mathbb{R}^k, \text{ et } \underline{x}^* \in S\} \subset B \tag{2.11}$$

C'est-à-dire que pour chaque $\underline{x}^* \in S$, son élément correspondant $[z_1(\underline{x}^*), \ldots, z_k(\underline{x}^*)]$, appartient à l'ensemble d'objectifs efficaces Z.

2.5. Solution préférée

Connue ainsi comme solution de compromis, est une solution efficace qui est choisie par le décideur comme la solution finale.

2.6. Représentation graphique d'un problème de PLMO.

Pour clarifier les concepts des solutions données ci-dessus, on va représenter graphiquement [cfr Figures (2.1) et (2.2)] le problème suivant [12] :

$$\max z(\underline{x}) = [z1(\underline{x}),\ z2(\underline{x})]$$

où : $z1(\underline{x}) = x1 - 3\,x2$

$z2(\underline{x}) = -4\,x1 + x2$

s.c. $g1(\underline{x}) = -x1 + x2 - 7/2 \leqslant 0$

$g2(\underline{x}) = x1 + x2 - 11/2 \leqslant 0$

$g3(\underline{x}) = 2\,x1 + x2 - 9 \leqslant 0$

$g4(\underline{x}) = x1 - 4 \leqslant 0$

$x1, x2 \geqslant 0$

Pour identifier S, il faut prendre l'image des points de l'espace d'actions [x1,x2] dans l'espace d'objectifs [z1,z2], utilisant les objectifs $z1(\underline{x})$ et $z2(\underline{x})$, et finalement appliquer la définition (2.9).

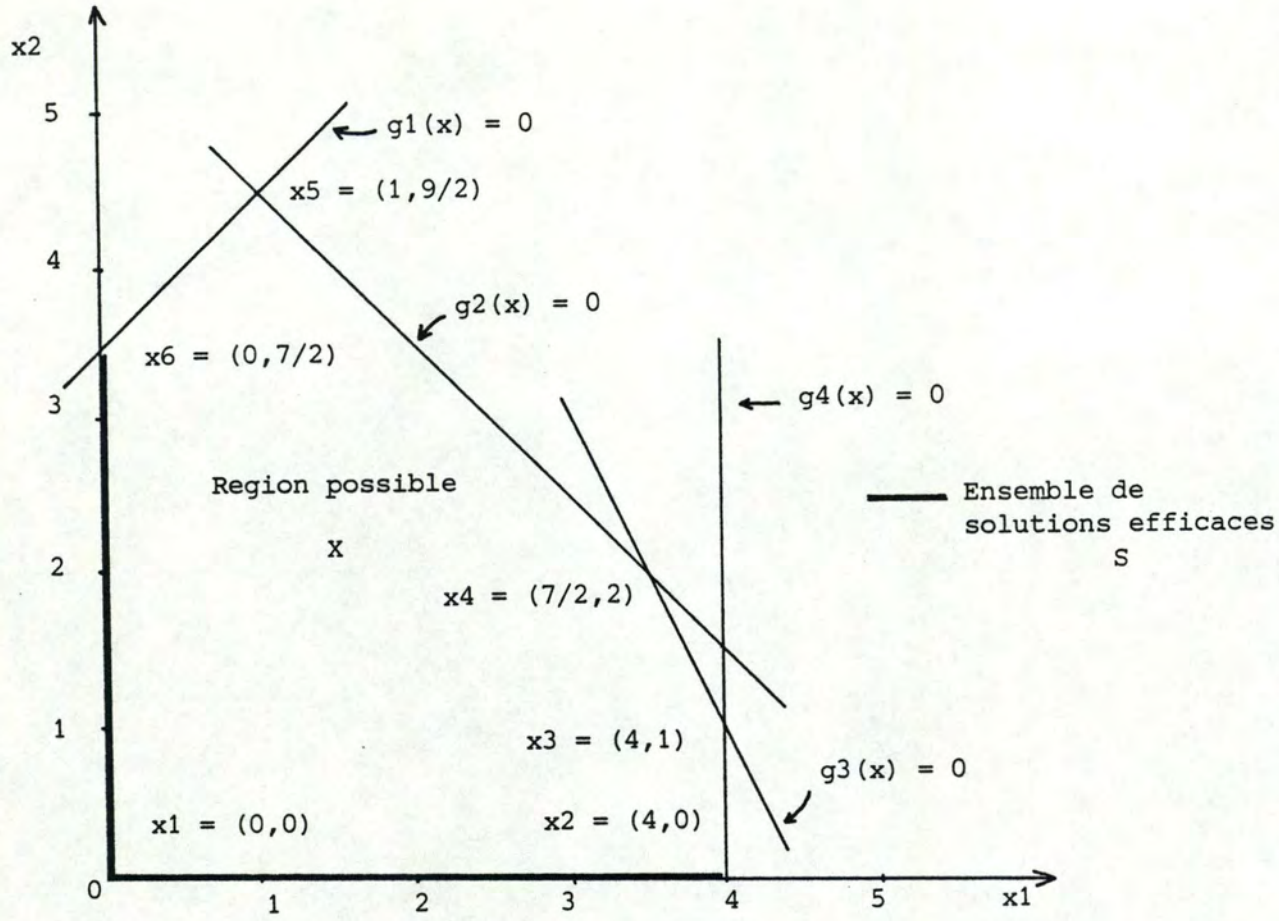


Figure 2.1. Region possible X et ensemble de solutions efficaces S dans l'espace d'actions.

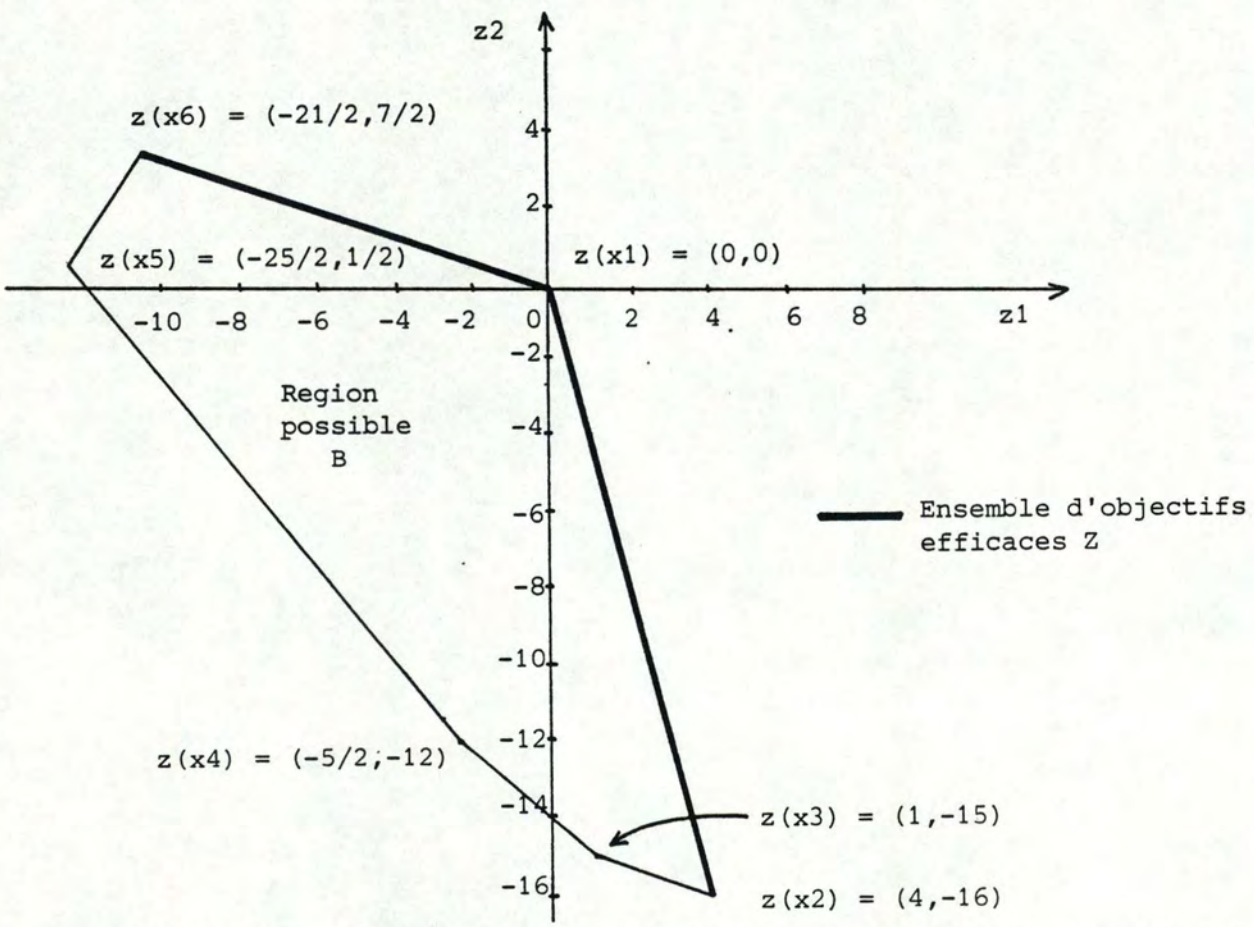


Figure 2.2. Region possible B et ensemble d'objectifs efficaces Z dans l'espace d'objectifs.

2.7. Fonction d'utilité Globale

La majorité des méthodes de programmation multiobjectifs utilisent la technique de modélisation de préférences où les objectifs jouent le rôle de critères de comparaison, pour pouvoir réunir les différents objectifs dans une seule fonction connue comme fonction d'utilité globale ($U(\underline{z})$). De cette manière, le problème de PLMO est ramené à un problème de programmation simple (un seul objectif).

$$\max U(\underline{z}) = U[z1(\underline{x}),\ldots,zk(\underline{x})] \qquad (2.12)$$
$$\underline{x} \in X$$

La fonction d'utilité globale peut être construite de plusieurs manières; la plus commune suppose que la fonction d'utilité est additivement séparable, c'est-à-dire qu'on peut l'écrire :

$$U(\underline{z}) = U1(z1(\underline{x})) + U2(z2(\underline{x})) +\ldots+ Uk(zk(\underline{x})) \qquad (2.13)$$

Chacune des fonctions de Uj est appelée contribution de l'objectif zj à l'utilité additive $U(\underline{z})$.

Si $\underline{z}^1 = (z_1^1, z_2^1, \ldots, z_k^1)$ et $\underline{z}^2 = (z_1^2, z_2^2, \ldots, z_k^2)$ sont deux vecteurs de Z, alors, $\underline{z}^1$ est préféré à $\underline{z}^2$ si $U(\underline{z}^1) > U(\underline{z}^2)$.

Un cas particulier d'utilité additive est celle de la somme pondérée, construite en utilisant des poids w pour indiquer l'importance relative de chaque objectif. Le problème de PLMO peut être représenté par

$$\max \sum_{j=1}^{k} wj\ zj(x) \qquad (2.14)$$
$$\underline{x} \in X$$

L'autre technique utilisée pour résoudre un problème de PLMO consiste à ranger les objectifs dans un ordre d'importance défini par le décideur.

On commence par maximiser l'objectif le plus important et on poursuit l'optimisation pour les objectifs suivants en accord avec l'ordre décidé.

Quand on va maximiser un des objectifs, les autres sont ajoutés à l'ensemble des contraintes pour s'assurer que la solution optimale satisfait un niveau acceptable prédéterminé pour ces objectifs.

2.8. Taux marginal de substitution

Une manière d'assigner les poids aux objectifs dans une fonction d'utilité additive est de demander au décideur les taux marginaux de substitution de chacun des objectifs par rapport à un objectif choisi comme objectif de référence.

Etant donné deux objectifs zi(x) et zj(x), on définit le taux marginal de substitution (trade-off) de l'objectif j par rapport à l'objectif i (objectif de référence) comme :

$$w_j^i = (\partial U/\partial zj) \,/\, (\partial U/\partial zi) \qquad (2.15)$$

Une manière d'obtenir les w_j^i est de déterminer la variation infinitésimale de l'objectif i (Δzi) par rapport à une variation Δzj de l'objectif j; c'est-à-dire :

$$w_j^i = - \frac{\Delta zi}{\Delta zj} \qquad (2.16)$$

Il signifie que, pour augmenter d'une unité la valeur de l'objectif zj, il faut diminuer la valeur de l'objectif de référence zi de w unités.

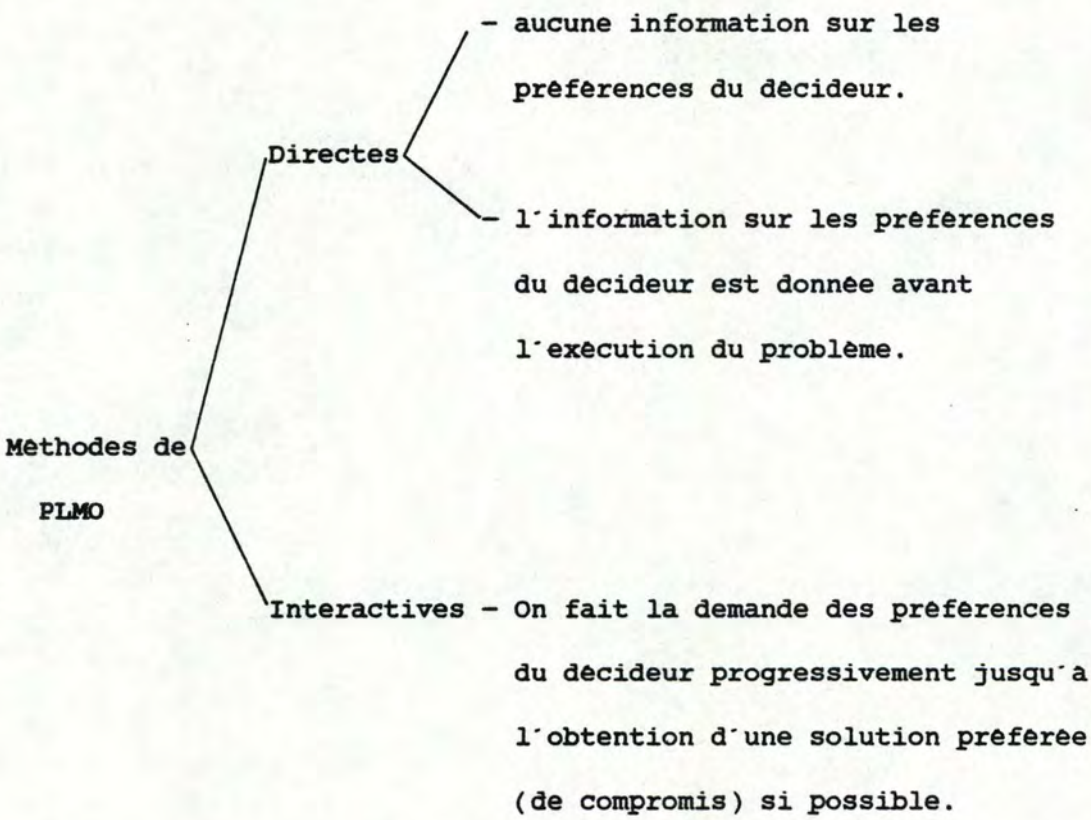
2.9. Décideur

Néologisme traduisant le mot anglais decision-maker (DM); un décideur est une personne individuelle ou un groupe de personnes qui intervient dans un processus de choix ou de décision.

Chapitre 3: METHODES DE PROGRAMMATION LINEAIRE MULTIOBJECTIFS

3.1. Types de méthodes

Les méthodes de PLMO peuvent être classées suivant leur caractère interactif avec le décideur quand elles lui demandent certaines informations sur ses préférences.



Actuellement, les méthodes interactives sont les plus répandues; elles permettent au décideur de maîtriser les caractéristiques du problème au fur et à mesure de son exécution (généralement, au début, le décideur ne peut donner que peu d'informations de préférence, vu la complexité du problème), et à chaque phase de dialogue, il pourra structurer d'une meilleure façon ces préférences. En plus comme indiqué dans [Chap. 1], la programmation mathématique, dans le contexte de problèmes multiobjectifs, sert uniquement comme "aide" au décideur pour qu'il puisse aboutir à une solution de compromis; elle ne sert jamais à sélectionner une solution.

La meilleure façon d'aider le décideur sera précisément de lui donner la possibilité d'obtenir la plus grande quantité de renseignements sur les effets de ses décisions et sur l'ensemble de solutions efficaces au problème.

La caractéristique commune à toutes les méthodes, pour la résolution de problèmes linéaires multiobjectifs, est une fonction objectif générée suivant différents critères; on se ramène de cette façon au cas de la programmation linéaire simple (un seul objectif) et par conséquent, à l'utilisation de l'algorithme du simplexe pour l'optimisation.

Il faut alors remarquer que, si on désire qu'une méthode de PLMO soit efficace et d'utilisation facile pour des problèmes réels, il est nécessaire de disposer d'un bon logiciel de PLS comme outil de base. Ce logiciel, en plus des méthodes pour résoudre un problème de PLS, devrait fournir l'information nécessaire et la possibilité de réaliser l'analyse de sensibilité, la paramétrisation, et garder au moins la dernière base du problème.

Puisque les méthodes du premier type (directes) seront difficilement appliquées dans des problèmes pratiques, on se limitera a présenter uniquement, comme exemple, la méthode du critère global.

Pour le deuxième type (méthodes interactives), on réalisera l'étude des méthodes suivantes: Stem, Geoffrion, Goal Programming Interactif, et Zionts-Wallenieus.

Toutes les méthodes étudiées ont été implémentées.

3.2. LA METHODE DU CRITERE GLOBAL

Cette méthode est basée sur la minimisation de la fonction objectif globale représentée par la somme des écarts relatifs entre les valeurs idéales des objectifs et ses valeurs actuelles.

$$\min \sum_{i=1}^{k} \frac{zi(\hat{\underline{x}}) - zi(\underline{x})}{zi(\hat{\underline{x}})} \qquad (3.1)$$

$$\text{s.c.} \quad \underline{x} \in X$$

Les solutions idéales $zi(\hat{\underline{x}})$, $i = 1,2,.....,k$
sont trouvées par l'optimisation des k problèmes suivants

$$\max zi(\underline{x}) \qquad (3.2)$$

$$\text{s.c.} \quad \underline{x} \in X$$

Bien que la méthode peut fournir une solution efficace, il est très probable qu'il y aura quelques objectifs comprenant des valeurs inacceptables pour le décideur.

On peut utiliser cette méthode, soit pour connaître les solutions idéales des objectifs, soit pour générer une solution initiale nécessaire pour les autres méthodes.

3.2.1. L'algorithme

#1. Pour chaque i = 1,2,...,k, calculer la solution optimale $\hat{\underline{x}}i$ du problème (3.2)

#2. Déterminer zi ($\hat{\underline{x}}i$), i = 1,2,...k

#3. Optimiser le problème (3.1) pour trouver la solution efficace $\underline{x}^*$;

Fournir $\underline{x}^*$;

STOP

3.3. LA METHODE DE GEOFFRION

Cette méthode suppose que le décideur possède, au moins implicitement, une fonction d'utilité globale. Le problème multiobjectif sera donc formulé de la façon suivante :

$$\max U\,[z1(\underline{x}),\ldots,zk(\underline{x})]$$

$$\text{s.c. } \underline{x} \in X \qquad (3.3)$$

La fonction d'utilité U[z(x)] n'est pas connue explicitement, et la procédure établira un dialogue avec le décideur, de manière à réunir une information équivalente à la fonction U (localement) et ainsi réaliser les calculs.

La méthode de Geoffrion suit l'algorithme de Frank-Wolfe [11], auquel on ajoute la démarche interactive avec le décideur.

Pour pouvoir résoudre (3.3) par l'algorithme de Frank-Wolfe, il faut émettre les hypothèses suivantes : a) la fonction d'utilité U est différentiable, continue et concave; b) la région possible X est convexe et compacte; c) chaque fonction objectif zi est concave; et, d) la fonction d'utilité U est croissante en chaque fonction objectif zi (parfois il faudra un changement de signe).

3.3.1. L'algorithme de Frank-Wolfe

Sans prendre en compte, pour le moment, le fait que U(z) n'est pas connue explicitement, l'algorithme de Frank-Wolfe appliqué à (2.3) consiste à :

#1. Choisir un point initial $\underline{x}^1 \in X$;

Fixer i=1

#2. a) Déterminer la solution optimale $\underline{y}^i$ du problème (3.4). Celle-ci permet de calculer la direction d'amélioration, au sens de la fonction d'utilité U issue du point $\underline{x}^i$

$$\max \quad \nabla_{\underline{x}} U[z1(\underline{x}^i),\ldots,zk(\underline{x}^i)].\underline{y}$$

$$\text{s.c.} \qquad \underline{y} \in X \tag{3.4}$$

b) Déterminer la "meilleure" direction $\underline{d}^i$ suivant laquelle il faut avancer pour améliorer la fonction d'utilité

$$\underline{d}^i = \underline{y}^i - \underline{x}^i \tag{3.5}$$

#3. a) Déterminer la quantité (t^i) à avancer le long de la meilleure direction définie dans #2.b, comme solution du problème d'optimisation:

$$\max U\ [z1(\underline{x}^i + t.\underline{d}^i),\ldots.,zk(\underline{x}^i+t.\underline{d}^i)]$$

$$\text{s.c.} \qquad 0 \leqslant t \leqslant 1 \tag{3.6}$$

b) Poser: $\underline{x}^{i+1} := \underline{x}^i + t^i.\underline{d}^i$;

Si $\underline{x}^{i+1} = \underline{x}^i$ alors

| STOP

sinon

| i:= i + 1;

| aller à #2

Puisque la fonction d'utilité U n'est pas connue

explicitement, la méthode de Geoffrion ajoute à l'algorithme de Frank-Wolfe l'approche interactive, ce qui permettra d'acquérir l'information de préférence du décideur pour construire la fonction U. De cette manière, les calculs en #2 et #3 peuvent être exécutés.

3.3.2. Détermination de la direction d'amélioration

Si on applique la règle de la chaîne à la fonction objectif de l'expression (3.4), celle-ci devient:

$$\nabla_{\underline{x}} U\,[z1(\underline{x}^i),.....,zk(\underline{x}^i)].\underline{y}^i$$

$$= \left[\frac{\partial U}{\partial z_1},.....,\frac{\partial U}{\partial z_k}\right]_{\underline{z}^i} \cdot \left[\begin{array}{ccc} \frac{\partial z_1}{\partial x_1} & ..... & \frac{\partial z_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial z_k}{\partial x_1} & ..... & \frac{\partial z_k}{\partial x_n} \end{array}\right]_{\underline{x}^i} \cdot \underline{y}^i$$

$$= \left[\sum_{j=1}^{k} \left[\frac{\partial U}{\partial z_j^i}\right] \cdot \nabla_{\underline{x}} z_j(\underline{x}^i)\right] \cdot \underline{y}^i \qquad (3.7)$$

où: $\dfrac{\partial U}{\partial z_j^i}$ est la dérivée partielle de U par rapport à l'objectif j, évaluée en le point $[z1(\underline{x}^i),.....,zk(\underline{x}^i)]$;

$\nabla_{\underline{x}} zj(\underline{x}^i)$ est le gradient de zj evalué en $\underline{x}^i$

La solution au problème (3.4) ne sera pas altérée, si on multiplie la fonction objectif par un scalaire; ainsi, on peut diviser la fonction objectif (3.7) par le coefficient constant $\dfrac{\partial U}{\partial z_1^i}$.

Le problème (3.4) qui permet de trouver la direction d'amélioration est équivalent à:

$$\max \sum_{j=1}^{k} w_j^i \nabla_{\underline{x}} zj(\underline{x}^i).\underline{y}^i$$

$$\text{s.c.} \quad \underline{y} \in X \qquad (3.8)$$

où w_j^i est définie comme:

$$w_j^i = (\partial U/\partial z_j^i)/(\partial U/\partial z_1^i) \ , \ j=1,2,\ldots,k \qquad (3.9)$$

Les w_j^i représentent les "taux marginaux de substitution" entre l'objectif j et un objectif de référence, arbitrairement choisi (dans notre cas on a choisi z1), en la solution actuelle $\underline{x}^i$. Les taux de substitution doivent être demandés au décideur, avant de résoudre le problème (3.8).

Une façon de calculer les w_j^i consiste à déterminer la variation
Δ z1 (du premier objectif) exactement compensée par une variation
Δ zj du jième objectif tandis que, les valeurs des autres objectifs restent constantes. L'équation (3.9) est approchée par:

$$w_j^i = -\frac{\Delta z_1^i}{\Delta z_j^i} \qquad (3.10)$$

3.3.3. Sélection de la distance(pas) d'amélioration

La solution au problème (3.6) peut être obtenue en présentant au décideur, sous forme d'un tableau, les valeurs des k objectifs $\underline{z}$ ($\underline{x}^i + t.\underline{z}^i$) calculés pour différentes valeurs de t comprises dans l'intervalle $0 \leqslant t \leqslant 1$. Le décideur détermine la valeur de t pour laquelle le vecteur d'objectifs z correspondant lui semble le plus satisfaisant; de cette manière, on a calculé la distance à avancer dans la direction d'amélioration.

La valeur de t choisie permet alors le calcul d'une nouvelle solution possible.

3.3.4. Critère de terminaison

Si en un point donné, le décideur trouve que la solution actuelle est la plus satisfaisante, il peut arrêter la procédure, soit en sélectionnant la valeur t = 0 (on ne peut plus obtenir d'amélioration), soit en signalant qu'il veut terminer les intéractions.

3.3.5. Analyse de la méthode

Le principal défaut de la méthode de Geoffrion est de considérer que le décideur est capable de fournir les taux de substitution entre deux objectifs quelconques, pour tous les objectifs, en n'importe quel point de la région possible définie par les contraintes du problème. Cette hypothèse est très forte et le décideur aura généralement des difficultés pour répondre aux questions qu'on lui pose.

Dyer [5] a proposé une procédure interactive pour aider le décideur à estimer les taux de substitution. Celle-ci est

basée sur les réactions du décideur face à une série de comparaisons ordinales par paire, qu'on lui propose. Le principe est le suivant: Supposons que la solution actuelle soit le point $\underline{x}^i$; on présente au décideur le vecteur

$$[z1(\underline{x}^i), z2(\underline{x}^i), \ldots, zj(\underline{x}^i), \ldots, zk(\underline{x}^i)]$$

nommé vecteur A;

et le vecteur

$$[z1(\underline{x}^i) + \Delta z_1^i, z2(\underline{x}^i), \ldots, zj(\underline{x}^i) - \Delta z_j^i, \ldots, zk(\underline{x}^i)]$$

nommé vecteur B.

Si le décideur préfère le vecteur B à A, on augmente Δz_j^i au double de sa valeur actuelle, et la question est répétée jusqu'à ce que le décideur préfère le vecteur A à B. A partir de cette sélection de préférence, l'algorithme de bissection est modifié pour calculer la valeur de Δz_j^i; si le décideur préfère A à B, on décroit Δz_j^i, et s'il préfère B à A on augmente Δz_j^i . Le processus interactif se poursuit jusqu'à ce qu'on trouve une valeur $(\Delta z_j^i)^*$ telle que le décideur soit indifférent face à A et B; le taux de substitution est alors défini comme:

$$w_j^i = -\frac{\Delta z_1^i}{(\Delta z_j^i)^*} \qquad (3.11)$$

La procédure est répétée pour calculer les taux de substitution pour tous les (k - 1) objectifs. Il faut noter ici qu'on a utilisé comme critère de référence, le premier objectif (c'est souvent le cas, sauf si le décideur désire en utiliser un autre); par convention on fixe $w_1^i = 1$; par tout i.

Les perturbations initiales Δz_1^i et Δz_j^i doivent être choisies de telle sorte qu'elles soient relativement petites par rapport aux valeurs des objectifs z_1^i et z_j^i respectivement, et à la fois suffisament grandes pour être significatives aux yeux du décideur.

Dans cette procédure de Dyer, il faut remarquer que les valeurs du vecteur B qui sont présentées au décideur ne correspondent pas nécessairement à des solutions possibles, ce qui peut induire le décideur à un faux jugement.

3.3.6. L'algorithme

#1. Choisir une solution initiale $\underline{x}^i$ (si possible efficace);

Calculer z_j^i $(\underline{x}^i)$, j=1,.....,k;

Poser i:=1

#2. Poser w_1^i:=1;

Pour chaque j=2,3,....,k calculer successivement les taux de substitution w_j^i des objectifs z_j^i par rapport à z_1^i

$$w = - \frac{\Delta z_1^i}{\Delta z_j^i}$$

en utilisant la procédure interactive de Dyer (3.2.5)

#3. Obtenir la solution $\underline{y}^i$ du problème de PLS

$$\max \sum_{j=1}^{k} w_j^i . \nabla_{\underline{x}} z_j(\underline{x}^i) \cdot \underline{y}$$

$$\text{s.c.} \quad \underline{y} \in X$$

#4. Calculer la direction d'amélioration de U, issue de $\underline{x}^i$

$$\underline{d}^i := \underline{y}^i - \underline{x}^i$$

#5. Poser t:=0;

Pour des intervalles de t, où $0 \leqslant t \leqslant 1$, Δt(par exemple Δt:=0.2), calculer les valeurs de $zj(\underline{x}^i + t.\underline{d}^i)$, j=1,2,.....,k; et élaborer le tableau suivant qui sera présenté au décideur

z \ t	t_0=0	$t_1 = \Delta t$	t_2=2 Δt	.....	t_η=1
z1	$z1(\underline{x}^i)$	$z1(\underline{x}^i+t_1.\underline{d}^i)$	$z1(\underline{x}^i+t_2.\underline{d}^i)$	.....	$z1(\underline{x}^i+\underline{d}^i)$
z2	$z2(\underline{x}^i)$	$z2(\underline{x}^i+t_1.\underline{d}^i)$	.....	.....	$z2(\underline{x}^i+\underline{d}^i)$
..	...	.....	.....	.....	.....
..					
zk	$zk(\underline{x}^i)$	$zk(\underline{x}^i+t_1.\underline{d}^i)$		.....	$zk(\underline{x}^i+\underline{d}^i)$

#6. D'après le tableau qu'on lui présente en #5, le décideur doit choisir la valeur t^i de t qui correspond au vecteur solution qui lui semble le plus satisfaisant.

#7. Si $t^i = 0$ alors

| STOP

sinon

| calculer $\underline{x}^{i+1} := \underline{x}^i + t^i.\underline{d}^i$;
| fournir la solution $\underline{x}^{i+1}$ et $\underline{z}(\underline{x}^{i+1})$;
| poser i:=i + 1;
| Si le décideur désire-t-il continuer les itérations alors
| | aller à #2
| sinon: STOP

3.4. GOAL PROGRAMMING INTERACTIF

La méthode Goal Programming permet au décideur de spécifier une cible, qu'il souhaite atteindre, pour chaque objectif. Une solution préférée est alors définie comme celle qui minimise la somme des écarts de l'ensemble des valeurs cibles.

Pour rendre la méthode plus efficace pour la résolution de problèmes multiobjectifs, on y a ajouté le dialogue avec le décideur afin de connaître l'information sur ses préférences; c'est ainsi qu'on parle plutôt dernièrement de Goal programming interactif.

3.4.1. Goal Programming

Dans l'approche Goal Programming, un problème multiobjectifs peut être formulé de la façon suivante:

$$\min \sum_{i=1}^{k} | zi(\underline{x}) - ci | \qquad (3.12)$$

$$\text{s.c.} \quad \underline{x} \in X$$

où: ci sont les cibles fixées par le décideur pour chaque objectif zi.

Le principe de la méthode est alors de minimiser la somme des valeurs absolues des écarts entre les valeurs cibles ci et les valeurs actuelles obtenues zi.

La fonction objectif dans (3.12) est non linéaire et on ne peut donc pas appliquer directement la méthode du simplexe; pour la ramener à une forme linéaire, on définit des nouvelles variables d'écart y_i^+ et y_i^- telles que:

$$y_i^+ = \frac{1}{2}\{|zi(\underline{x}) - ci| + [zi(\underline{x}) - ci]\} \quad (a)$$
$$y_i^- = \frac{1}{2}\{|zi(\underline{x}) - ci| - [zi(\underline{x}) - ci]\} \quad (b) \qquad (3.13)$$

où, y_i^+ est l'écart positif entre la ième cible et le ième objectif (dépassement de la valeur cible), et y_i^- est l'écart négatif entre la ième cible et le ième objectif (la valeur cible n'est pas atteinte).

Si on additionne (3.13.a) à (3.13.b) on obtient

$$y_i^+ + y_i^- = |zi(\underline{x}) - ci| \qquad (3.14)$$

Ainsi la fonction objectif dans (3.12) peut être simplifiée grâce à la relation linéaire équivalente (3.14). En plus, si on soustrait (3.13.a) de (3.13.b), on obtient

$$zi(\underline{x}) - ci = y_i^+ - y_i^- \qquad (3.15)$$

La formulation linéaire équivalente à (3.12) est

$$\min \sum_{i=1}^{k} (y_i^+ + y_i^-)$$

$$\text{s.c.} \quad \underline{x} \in X \qquad (3.16)$$

$$zi(\underline{x}) - y_i^+ + y_i^- = ci$$

$$y_i^+, y_i^- \geq 0, \; i=1,\ldots,k$$

Puisqu'il n'est pas possible de dépasser et de ne pas atteindre la cible en même temps, au moins une des deux variable d'écart doit avoir la valeur zéro, c'est-à-dire:

$$y_i^+ . y_i^- = 0.$$

Cette contrainte est vérifiée automatiquement par la méthode

du simplexe.

On peut éventuellement assigner des poids wi aux variables d'écart $yi(y_i^+, y_i^-)$, ce qui permettra de mesurer l'importance relative à attribuer à chacun des objectifs.

Finalement, la formulation Goal programming d'un problème de PLMO s'écrit

$$\min \sum_{i=1}^{k} wi.(y_i^+ + y_i^-)$$

$$\text{s.c.} \quad \underline{x} \in X \qquad (3.17)$$

$$zi(\underline{x}) - y_i^+ + y_i^- = ci$$

$$y_i^+, y_i^- \geq 0, \quad i=1,.....,k$$

3.4.2. Goal Programming Interactif

Si on suppose que la préférence du décideur pour l'obtention de ses cibles est non décroissante, c'est-à-dire que la minimisation des dépassements $\underline{y}^+$ n'est pas nécessaire, l'équation (3.17) devient

$$\min \sum_{i=1}^{k} wi.y_i^-$$

$$\text{s.c.} \quad \underline{x} \in X \qquad (3.18)$$

$$zi(\underline{x}) + y_i^- - y_i^+ = ci$$

$$y_i^+, y_i^- \geq 0, \quad i=1,.....,k$$

Cette formulation particulière est connue comme "one-sided" Goal programming, due à ce qu'on minimise seulement les écarts négatifs.

La fonction objectif dans (3.18) est une expression mathématique équivalente à celle du problème (3.3) définie par la méthode de Geoffrion. Les considérations pour rendre interactive la méthode Goal programming sont alors les mêmes que pour la méthode de Geoffrion. Il en est de même pour tous les calculs à effectuer.

Pour montrer la relation entre (3.18) et (3.3), il faut noter dans (3.18) que:

$$\underline{y}^- = \underline{c} - \underline{z}(\underline{x}) + \underline{y}^+$$

où le vecteur des cibles $\underline{c}$ (valeurs constantes), peut être ignoré puisque l'objectif est de minimiser $\underline{y}^-$; la fonction objectif de (3.18) devient

$$\min \underline{w}.[\ -\underline{z}(\underline{x}) + \underline{y}^+]$$

qui est équivalente à

$$\max \underline{w}.[\ \underline{z}(\underline{x}) - \underline{y}^+] \qquad (3.19)$$

La fonction objectif (3.19) est aussi un cas particulier de

$$\max U[\underline{z}(\underline{x})]$$

qui est la fonction objectif dans (3.3).

L'objectif défini en (3.19) est une fonction d'utilité U additive séparable, c'est-à-dire:

$$U[\underline{z}(\underline{x})] = U1[z1(\underline{x})]+.....+Uk[zk(\underline{x})],$$

avec un accroissement marginal d'utilité par rapport à une unité additionnelle de zj au-dessus de cj égale à zéro; c'est-à-dire,

$$\frac{\partial Uj}{\partial zj} = 0 \text{ pour } zj \geqslant cj \quad \text{et} \quad Uj[zj(\underline{x})] = wj.[zj(\underline{x}) - y_j^+\ ]$$

Etant montré que la formulation d'un problème de PLMO dans (3.18) est une autre expression mathématique de (3.3), la méthode goal programming interactif suit exactement l'algorithme de Geoffrion, et la modification consiste à remplacer (3.3) par (3.18).

3.4.3. L'algorithme

#1. Demander au décideur la valeur cible ci pour chaque fonction objectif zi.

#2. Choisir une solution initiale $\underline{x}^i$;

Calculer $z_j^i(\underline{x}^i)$, j=1,.....,k;

Poser i:=1

#3. Poser w_1^i:=1;

Pour chaque j=2,3,....,k calculer successivement les taux de substitution w_j^i des objectifs z_j^i par rapport à z_1^i

$$w_j^i = - \frac{\Delta z_1^i}{\Delta z_j^i}$$

en utilisant la procédure interactive de Dyer [3.2.5]

#4. Résoudre le problème "one-sided" goal programming

$$\min \sum_{j=1}^{k} w_j^i . y_j^-$$

$$\text{s.c.} \qquad \underline{u} \in X$$

$$z_j(\underline{u}) + y_j^- - y_j^+ = c_j$$

$$y_j^+, y_j^- \geq 0, \quad j=1,....,k$$

où: $\underline{u}^i$ sera la valeur associée à la solution optimale $(\underline{y}^-)^i$.

N.B. Le point $\underline{u}^i$ est l'équivalent du point $\underline{y}^i$ qu'on obtient dans #3 de l'algorithme de Geoffrion [3.2.6].

#5. Calculer $\underline{d}^i := \underline{u}^i - \underline{x}^i$

#6. Poser t:=0;

Pour des intervalles de t, où $0 \leq t \leq 1$, Δt(par exemple Δt:=0.2), calculer les valeurs de $z_j(\underline{x}^i + t.\underline{d}^i)$, j=1,2,.....,k; et élaborer le tableau suivant qui sera présenté au décideur:

z \ t	$t_0=0$	$t_1=\Delta t$	$t_2=2\,\Delta t$	.....	$t_n=1$
z1	$z1(\underline{x}^i)$	$z1(\underline{x}^i+t_1.\underline{d}^i)$	$z1(\underline{x}^i+t_2.\underline{d}^i)$	.....	$z1(\underline{x}^i+\underline{d}^i)$
z2	$z2(\underline{x}^i)$	$z2(\underline{x}^i+t_1.\underline{d}^i)$	.....	.....	$z2(\underline{x}^i+\underline{d}^i)$
..	...	.....	.....	.....	.....
..					
zk	$zk(\underline{x}^i)$	$zk(\underline{x}^i+t_1.\underline{d}^i)$		.....	$zk(\underline{x}^i+\underline{d}^i)$

#7. D'après le tableau qu'on lui présente en #5, le décideur doit choisir la valeur t de t qui correspond au vecteur solution qui lui semble le plus satisfaisant

#8. Si $t^i = 0$ alors

    STOP

sinon

    calculer $\underline{x}^{i+1} := \underline{x}^i + t^i.\underline{d}^i$;

    fournir la solution $\underline{x}^{i+1}$ et $z(\underline{x}^{i+1})$;

    poser i:=i + 1;

    Si le décideur désire-t-il continuer les itérations alors

        aller à #2

    sinon: STOP

3.5. LA METHODE STEM

Souvent, dans un problème de PLMO, le décideur est incapable de fournir, à priori, l'information sur l'importance relative de chaque fonction objectif, ainsi que les taux de substitutions, ce qui permettrait de construire une fonction d'utilité globale.

La méthode Stem est une procédure itérative d'exploration des solutions ne faisant pas référence, même implicitement, à une fonction d'utilité. Le "meilleur compromis" est obtenu après un nombre d'itérations au maximum égal au nombre de fonctions objectif du problème.

Chaque itération comprend une phase de calcul, suivie d'une phase de décision; pendant la phase de décision, le décideur analyse les résultats fournis par la phase de calcul et donne de

nouvelles informations sur ses objectifs.

3.5.1. Phase de calcul

Elle consiste en la construction d'un tableau connu comme "tableau de gains" (pay-off), et calcule une solution efficace $\underline{x}^*$ qui est proposée au décideur.

3.5.1.1. Construction du tableau de gains

Ce tableau est construit une seule fois avant la première itération; pour l'établir, on doit d'abord résoudre les k problèmes suivants

$$\max zj(\underline{x}) = \sum_{i=1}^{n} cij.xi \ , \ j=1,2,....,k$$

$$\text{s.c.} \qquad \underline{x} \in X \tag{3.20}$$

Le tableau des gains aura la configuration suivante

	z1	z2	...	zj	...	zk
z1	M1	z_2^1	...	z_j^1	...	z_k^1
z2	z_1^2	M2	...	z_j^2	...	z_k^2
.	.	.	...	.	...	.
zj	z_1^j	z_2^j	...	Mj	...	z_k^j
.	.	.	...	.	...	.
zk	z_1^k	z_2^k	...	z_j^k	...	Mk

Tableau 3.1

Dans le tableau 3.1, la ligne j contient les valeurs des objectifs z1,z2,....,zk pour la solution $\underline{x}^j$ qui maximise l'objectif zj selon (3.20); ainsi z_j^i est la valeur de l'objectif zj quand l'objectif zi atteind son maximun. On note $Mj = z_j^j$, j=1,2,....,k , les valeurs optimales des objectifs.

La solution idéale $\hat{\underline{x}}$ à laquelle correspond le vecteur $\hat{\underline{z}}$ = [M1,M2,.....,Mk], c'est-à-dire les valeurs de la diagonale du tableau 3.1, n'est presque jamais obtenue [cfr 2.3], et se trouve généralement en dehors de la

région possible. Ici, elle est pourtant utilisée comme référence pour évaluer les solutions efficaces.

3.5.1.2. Calcul d'une solution efficace

A chaque itération r, on cherche une solution efficace $\underline{x}^r$, qui soit la "plus proche", dans le sens minimax, de la solution idéale $\hat{\underline{x}}$.

On résout donc le problème linéaire suivant:

$$\begin{aligned} &\min \lambda \\ \text{s.c.} \quad &\lambda \geq \{Mj - zj(\underline{x})\}.\Pi_j \ , \ j=1,.....,k \\ &\underline{x} \in X^r \\ &\lambda \geq 0 \end{aligned} \tag{3.21}$$

où: λ représente l'écart maximum pondéré d'un objectif de la solution idéale, et Π_j sont les poids donnant l'importance relative des écarts aux valeurs optimales.

L'ensemble X^r dans (3.21) consiste en la région possible originale X (où, r=1), plus toute contrainte ajoutée à la (r - 1) ième itération; $\underline{x}^r \in X$ représente la solution optimale au problème (3.21).

Les poids relatifs des écarts sont définis de la façon suivante:

$$\Pi_j = \frac{\alpha_j}{\sum_{i=1}^{k} \alpha_i} \tag{3.22}$$

où: $$\alpha_j = \frac{|Mj - mj|}{Max(|Mj|, |mj|)} \cdot \frac{1}{\sqrt{\sum_{i=1}^{n} (cij)^2}} \tag{3.23}$$

Dans (3.23), mj est la valeur minimale de la colonne j dans le tableau des gains 3.1, et Mj sa valeur maximale.

D'après le premier terme de (3.23), on peut conclure que, si toutes les valeurs $z_i^{\sim}$, i=1,..,k (i≠j) sont "proches" de la valeur optimale M_j, alors l'objectif zj n'est pas sensible aux variations des valeurs de pondération et on assignera un poids Π_j faible à l'objectif zj; par contre, si les valeurs $z^{\sim}$ sont très distinctes de celles des Mj, on assignera à zj un poids Π_j fort.

Le deuxième terme de (3.23) permet de normer les valeurs des fonctions objectif.

Les α_j sont utilisés pour définir les poids Π_j de telle façon que leur somme soit égale à 1; cela signifie que différentes solutions, obtenues par l'application de différentes stratégies de pondération, peuvent être comparées facilement.

3.5.2. Phase de décision

La solution efficace $\underline{x}^r$ obtenue dans la phase de calcul, est proposée au décideur, qui, en la rapprochant de la solution idéale $\hat{\underline{x}}$, décide si oui ou non elle lui convient. Si $\underline{x}^r$ lui convient, c'est la solution de compromis, et la procédure est terminée; sinon, il doit accepter de relâcher un objectif pour essayer d'améliorer les autres: il indique alors l'objectif z_j^* et la quantité à relâcher (Δz_j^*).

A l'itération suivante, la région possible est modifiée et devient

$$X^{r+1} = \begin{cases} X^r \\ z_j^*(x) \geq z_j^*(\underline{x}^r) - \Delta z_j^* \\ zi(\underline{x}) \geq zi(\underline{x}^r),\ i=1,2,\ldots,k\ ;\ i\neq j \end{cases} \qquad (3.24)$$

Le poids Π_j^* est fixé à la valeur zéro, et on retourne à la phase de calcul.

La procédure s'arrête lorsque la solution proposée au décideur lui convient, ou en tous cas, après au plus k itérations, parce que la fonction objectif qu'on a relâchée n'est plus considérée pour cet effet dans les itérations postérieures ($\Pi_j^* = 0$).

3.5.3. Analyse de la méthode

- L'information qu'on demande au décideur, dans la phase de dialogue, n'est pas compliquée. De plus, la méthode garantit la convergence en k itérations au plus.

  Cependant, les décisions prises par le décideur sont irrévocables. En effet, puisqu'il ne connaît pas d'avance les effets que subiront les autres objectifs par le relâchement Δz_j^* de l'objectif z_j^*, et s'il s'aperçoit à l'itération suivante qu'il a trop (ou trop peu) relâché l'objectif z_j^*, il n'a plus la possibilité de revenir en arrière.

  Vincke [cfr 3.4.5] propose une variante à la méthode Stem, basée sur l'analyse de sensibilité d'un problème de PLS, qui permet de fournir au décideur les conséquences des variations qu'il propose.

- La solution présentée par la méthode lors de la phase de calcul, n'est pas nécessairement efficace; il peut exister une solution possible donnant la même valeur optimale à λ, et dominant la solution trouvée par la procédure.

3.5.4. L'algorithme

#1. Pour chaque j=1,....,k successivement, calculer les solutions optimales $\hat{x}j$ des problèmes

$$\max zj(\underline{x}) = \sum_{i=1}^{n} cij.xi \ , \ j=1,2,....,k$$

$$\text{s.c.} \quad \underline{x} \in X$$

#2. Construire le tableau des gains; pour chaque $\hat{x}j$, calculer les éléments de la colonne i du tableau:

$$z_j^i = zj(\underline{\hat{x}}i), \ i=1,2,.....,k$$

#3. Pour chaque colonne i du tableau, i=1,2,...,k

déterminer sa valeur minimun mi

#4. Effectuer: $X^1 := X$;

$r := 1$

#5. Calculer

$$\alpha_j = \frac{|Mj - mj|}{\text{Max}(|Mj|, |mj|)} \cdot \frac{1}{\sqrt{\sum_{i=1}^{n} (cij)^2}} \quad , \; j=1,....k;$$

#6. Calculer

$$\Pi_j = \frac{\alpha_j}{\sum_{i=1}^{k} \alpha_i} \quad , \; j=1,.....,k$$

#7. Trouver la solution x du problème de PLS

$$\begin{aligned} &\min \ \lambda \\ \text{s.c.} \quad &\underline{x} \in X^r \\ &\lambda \geq \{Mj - zj(\underline{x})\}.\Pi_j \ , \; j=1,....,k \\ &\lambda \geq 0 \end{aligned}$$

#8. Si r=k alors

| fournir $\underline{x}^r$; STOP

sinon: passer à l'étape suivante

#9. Présenter au décideur les vecteurs $\hat{\underline{z}}$ et $\underline{z}^r$;

Si le vecteur $\underline{z}^r = [z1(\underline{x}^r),.....,zk(\underline{x}^r)]$ convient-il au décideur alors

| fournir $\underline{x}^r$; STOP

sinon

| lui demander l'objectif z_j^* à relâcher;

| lui demander la quantité Δz_j^* de relâchement

#10. Effectuer:

$$\alpha_j^* := 0;$$

$$x^{r+1} = \begin{cases} x^r \\ z_j^*(x) \geq z_j^*(\underline{x}^r) - \Delta z_j^* \\ zi(\underline{x}) \geq zi(\underline{x}^r),\ i=1,2,\ldots,k\ ;\ i \neq j \end{cases}$$

r:= r + 1;

aller a #6

3.5.5. La variante de Vincke

Dans cette approche, après le calcul de la première solution efficace par la méthode STEM, on établit un dialogue avec le décideur basé sur les propriétés de l'algorithme du simplexe et sur l'analyse de sensibilité d'un PLS; de cette manière, on permet au décideur de connaître les conséquences des modifications qu'il propose.

Pour pouvoir réaliser l'analyse de sensibilité sur les valeurs des objectifs, on ajoute à l'ensemble des contraintes du problème (3.21) les fonctions objectifs et le problème devient:

$$\min \ \lambda$$

$$\text{s.c} \qquad zj = \sum_{i=1}^{n} cij.xj \ ,\ j=1,\ldots,k$$

$$\lambda \geq \{Mj - zj(\underline{x})\}. \ ,j=1,\ldots,k \qquad (3.25)$$

$$\underline{x} \in x^r$$

$$\lambda \geq 0$$

Lorsqu'on a calculé la première solution efficace avec STEM, la partie interactive est basée sur l'analyse de sensibilité des termes indépendants (RHS) des contraintes du problème (3.25); ceci permet de réaliser l'étude de n'importe quelle variation proposée par le décideur. Les solutions à ces variations seront obtenues avec une simple lecture du dernier tableau du simplexe, après un eventuel changement de base par l'application de l'algorithme Dual du simplexe. La

procédure s'arrête quand le décideur trouve que la solution qu'on lui propose est satisfaisante.

Dans cette procédure, aucune solution n'est rejetée définitivement, puisque on peut revenir en arrière.

3.6. LA METHODE DE ZIONTS ET WALLENIUS

Dans cette méthode, les préférences du décideur sont représentées par une "fonction d'utilité" globale et on suppose les hypothèses suivantes: a) toutes les fonctions objectif du problème sont concaves (à maximiser), b) les contraintes forment un ensemble convexe, c) la fonction d'utilité globale n'est pas connue explicitement par le décideur, mais elle est implicitement une fonction linéaire, et plus généralement, une fonction concave.

La méthode consiste à choisir initialement un ensemble de multiplicateurs (poids) positifs et à générer la fonction d'utilité en utilisant ces multiplicateurs. La fonction d'utilité optimisée, on obtient une solution efficace du problème; à partir de l'ensemble de variables hors-base, on sélectionne un sous-ensemble de variables hors-base efficaces. Un ensemble de taux de substitution (variations) des objectifs est défini pour chacune des variables hors-base efficaces. Dans une phase de décision, elles seront présentées au décideur qui doit indiquer pour chaque ensemble de variations s'il lui convient, ne lui convient pas, ou, s'il y est indifférent. A partir des réponses du décideur, on détermine un nouvel ensemble de multiplicateurs et la solution efficace associée. On répète alors la procédure et un nouvel ensemble de variations est présenté au décideur; la convergence de la méthode est assurée si les préférences du décideur sont toujours cohérentes.

3.6.1. Génération et optimisation de la fonction d'utilité

La fonction d'utilité est générée par un ensemble de multiplicateurs (choisis initialement arbitrairement); une solution efficace est obtenue par la résolution du problème de PLS suivant:

$$\max U(x) = \sum_{i=1}^{k} \lambda_i . zi(\underline{x})$$

$$\text{s.c.} \quad \underline{x} \in X \qquad (3.26)$$

$$\sum_{i=1}^{k} \lambda_i = 1$$

$$\lambda_i \geq 0, \; i=1,\ldots,k$$

Puisque U et les contraintes du problème sont linéaires, la

Puisque U et les contraintes du problème sont linéaires, la solution efficace obtenue dans (3.26) sera nécessairement un point extrémal du convexe X.

3.6.2. Détermination des variables hors-base efficaces

Pour chaque variable hors-base xr correspondant à la solution efficace trouvée dans la phase précédente, on teste si l'introduction de xr dans la base produit simultanément une augmentation d'un objectif et la diminution d'au moins un des autres objectifs; dans l'affirmative, xr est appelée variable "hors-base efficace". Pour déterminer si une variable hors-base xr, xr$\in$ N, est efficace, il faut d'abord résoudre le problème suivant:

$$\min \sum_{i=1}^{k} wir.\lambda_i$$

$$\text{s.c.} \quad \sum_{i=1}^{k} wij.\lambda_i \geq 0, \; j \in N; \; j \neq r \qquad (3.27)$$

$$\sum_{i=1}^{k} \lambda_i = 1$$

$$\lambda_i \geq 0, \; i=1,.....,k$$

où: N est l'ensemble de variables hors-base.

wij sont les variations (>0, ou <0) de chaque fonction objectif zi dues à l'introduction d'une unité de la variable hors-base xj dans la solution.

Une façon d'obtenir ces valeurs wij est d'ajouter les fonctions objectif à l'ensemble de contraintes, ce qui permet de traiter les objectifs zi comme des variables dans le problème de PLS

$$zi = \sum_{j=1}^{n} cij.xj, \; i=1,....,k$$

Les valeurs de wij seront alors obtenues directement à partir du tableau simplexe final.

Les variations wij pour une variable hors-base xj sont connues aussi, inadéquatement, comme "taux marginaux de substitution".

Une fois le problème (3.27) optimisé, si la valeur minimum de la fonction objectif fr est négative, la variable hors-base xr est efficace; au contraire si elle est positive, la variable xr n'est pas efficace. Si l'ensemble de variables hors-base efficaces est vide, la procédure s'arrête.

Remarque: Pour qu'une variable hors-base xj soit efficace, il doit y avoir au moins une valeur wij positive et au moins une valeur wij négative; si toutes les valeurs wij pour la variable xj sont positives, celle-ci ne peut être une variable efficace et il n'est pas nécessaire de résoudre (3.27) pour cette variable.

3.6.3. Phase de décision

Pour chaque variable hors-base efficace xr, on pose au décideur la question suivante: "Etes-vous prêt à accepter une variation de w1r pour l'objectif z1, une variation de w2r pour l'objectif z2,....,et une variation de wkr pour l'objectif zr?. Répondez oui, non, ou indifférent à cette proposition".

Si il répond "non" pour toutes les variables hors-base efficaces, la procédure s'arrête; sinon on construit des contraintes, pour restreindre la sélection des multiplicateurs qui seront utilisés pour trouver une nouvelle solution efficace, de la façon suivante:

- Pour chaque réponse affirmative, on introduit une inégalité de la forme

$$\sum_{i=1}^{k} wir.\lambda_i \leqslant -\varepsilon \qquad (3.28)$$

où est une quantité positive arbitrairement petite.

Le décideur veut que la contribution unitaire de xr à sa fonction d'utilité soit positive.

- Pour chaque réponse négative, on introduit une inégalité de la forme:

$$\sum_{i=1}^{k} wir.\lambda_i \geqslant \varepsilon \qquad (3.29)$$

- Pour chaque réponse d'indifférence, on introduit une égalité de la forme:

$$\sum_{i=1}^{k} wir.\lambda_i = 0 \qquad (3.30)$$

En pratique, les réponses d'indifférence ne sont pas considérées car elles impliquent un degré trop élevé de précision dans le jugement par le décideur.

3.6.4. Détermination d'un nouvel ensemble de multiplicateurs

Un nouvel ensemble de multiplicateurs (poids) est déterminé par l'obtention d'une solution possible à l'ensemble de contraintes constitué par les équations (3.28), (3.29) et,

$$\sum_{i=1}^{k} \lambda_i = 1$$

$$\lambda_i \geq \varepsilon \,, \quad i=1,\ldots,k \qquad (3.31)$$

Cet ensemble sera utilisé pour générer la fonction d'utilité et trouver une autre solution efficace du problème, qui sera toujours adjacente à la précédente.

Chaque itération de la méthode utilise les contraintes (3.28) et (3.29) qui ont été définies dans les itérations précédentes, plus celles qui sont ajoutées par l'itération courante.

3.6.5. Analyse de la méthode

- La procédure fournit comme solutions efficaces uniquement les points extrêmaux du polyèdre de l'ensemble de solutions possibles, restreignant considérablement l'ensemble des solutions efficaces. On ne présente pas au décideur les solutions qui se trouvent dans les segments reliant les points extrêmaux, et qui seraient, très probablement, parmis les préférées. L'aide qu'on donne au décideur pour aboutir à une solution de compromis, est très limitée.

- L'information qu'on demande au décideur sur ses préférences est très complexe, malgré les arguments des auteurs pour montrer la simplicité des questions posées au décideur. En effet, pour un problème qui possède une dizaine de contraintes, les questions posées au décideur seront très nombreuses et le décideur pourra très difficilement rester cohérent dans ses réponses.

- Le nombre de programmes linéaires à résoudre est assez important; il dépend du nombre de contraintes du problème.

- Généralement, la méthode faussera les préférences du décideur; le fait que le décideur accepte les variations dans les valeurs des objectifs dues à une augmentation d'une unité de la variable hors-base efficace xr (c'est-à-dire qu'il accepte un petit déplacement sur l'arête du polyèdre), n'implique pas qu'il accepte des variations telles que le déplacement jusqu'au sommet voisin.

3.6.6. L'algorithme

#1. Choisir un ensemble de multiplicateurs λ^1, par exemple:

$$\lambda^1_i = \frac{1}{k}, \quad i=1,\ldots,k \ ;$$

Poser p:= 1 ;

Choisir une valeur ε petite (ex: $\varepsilon = 10^{-3}$)

#2. Calculer la solution efficace $\underline{x}^p$ par l'optimisation du problème:

$$\max U(\underline{x}) = \sum_{i=1}^{k} \lambda^p_i . zi(\underline{x})$$

$$\text{s.c.} \qquad \underline{x} \in X$$

$$zi = \sum_{j=1}^{n} cij.xj \ , \quad i=1,\ldots,k$$

$$\sum_{i=1}^{k} \lambda^p_i = 1$$

$$\underline{\lambda}^p \geq \underline{0}$$

#3. Fournir $\underline{x}^p$ et $\underline{z}(\underline{x}^p)$;

Définir N = {variables hors-base}, dans la solution $\underline{x}^p$;

Pur chaque x_r , $x_r \in N$, déterminer (par lecture du dernier tableau simplexe) son ensemble de variations wir, i=1,....,k ;

$\forall x_r$: Si tous les wir > 0 alors

$$N = N \setminus x_r$$

sinon: aller à #4

#4. Poser Nl = $\{\phi\}$; (Nl : ensemble de variables hors-base efficaces) Pour chaque $x_r \in N$, optimiser le problème de PLS suivant:

$$\min fr = \sum_{i=1}^{k} wir.\lambda_i$$

$$s.c. \quad \sum_{i=1}^{k} wij.\lambda_i \geq 0, \quad j \in N; \; j \neq r$$

$$\sum_{i=1}^{k} \lambda_i = 1$$

$$\underline{\lambda} \geq \underline{0}$$

Si fr < 0 alors: Nl = Nl U $\{x_r\}$

#5. Poser cra := 0; (cra: compteur de réponses affirmatives)

Si Nl = $\{\phi\}$ alors

STOP

sinon: aller à #6

#6. Pour chaque xr $\in$ Nl, présenter au décideur l'ensemble des variations wir et lui demander s'il l'accepte ou pas

si réponse affirmative: construire la contrainte

$$\sum_{i=1}^{k} wir.\lambda_i \leq -\varepsilon \; ;$$

calculer cra := cra + 1

si réponse négative: construire la contrainte

$$\sum_{i=1}^{k} wir.\lambda_i \geq \varepsilon$$

#7. Si cra = 0 alors

    STOP

sinon

- en utilisant toutes les contraintes définies dans #6 au cours des itérations précédentes et par l'actuelle (avec en plus la contrainte (3.21)), calculer un nouvel ensemble de poids, par la résolution du programme linéaire généré par les contraintes signalées ci-dessus ;

- poser p := p + 1 ;

- aller à #2

Chapitre 4: LE LOGICIEL DE PROGRAMMATION LINEAIRE LAMPS

4.1. Introduction

Toutes les méthodes étudiées dans le chapitre précédent utilisent l'optimisation de sous-problèmes qui sont des programmes linéaires. En plus, certaines d'entre elles se servent des caractéristiques de l'algorithme du simplexe pour obtenir des informations dont elles ont besoin pour l'application de leurs algorithmes.

Jusqu'à présent, dans le cadre de la programmation mathématique, on a considérablement développé des logiciels concernant la résolution de problèmes de programmation linéaire simple, et on trouve généralement ces logiciels offerts par les systèmes informatiques comme partie du système opérationnel (programmes utilitaires). Pour implanter un logiciel de programmation linéaire multiobjectifs, on peut (plutôt on doit) utiliser un logiciel de programmation linéaire comme outil de base, pour autant qu'il soit disponible. La performance d'une méthode de PLMO va dépendre fortement de la performance et des facilités fournies par le logiciel de base utilisé. Normalement, les caractéristiques qui déterminent la qualité d'un logiciel de PLS sont : la gestion des entrées-sorties, la richesse des algorithmes, le temps d'exécution des calculs, et les possibilités de réaliser les analyses de post-optimisation.

En ce qui concerne ce travail, nous avons utilisé le logiciel de programmation linéaire simple LAMPS, disponible sur le système DEC-20 de l'Institut d'Informatique, sur lequel a été greffé le système de PLMO à développer.

4.2. Caractéristiques du logiciel LAMPS

LAMPS peut être utilisé en mode interactif ou en mode batch (comme une série de sous-routines). Initialement, il a été développé pour travailler en mode interactif uniquement; pour chaque module du logiciel, il existe une commande qui permet de spécifier les paramètres obligatoires et facultatifs que le module nécessite pour son exécution.
Actuellement, la presque totalité des modules du logiciel LAMPS peuvent être utilisés en tant que sous-routines par des programmes externes à LAMPS.

La configuration du LAMPS pour la programmation linéaire simple (il y a aussi des algorithmes pour la résolution de problèmes de programmation linéaire en nombres entiers) est celle schématisée à la figure (4.1). Les noms placés sur les axes sont ceux des différents modules.

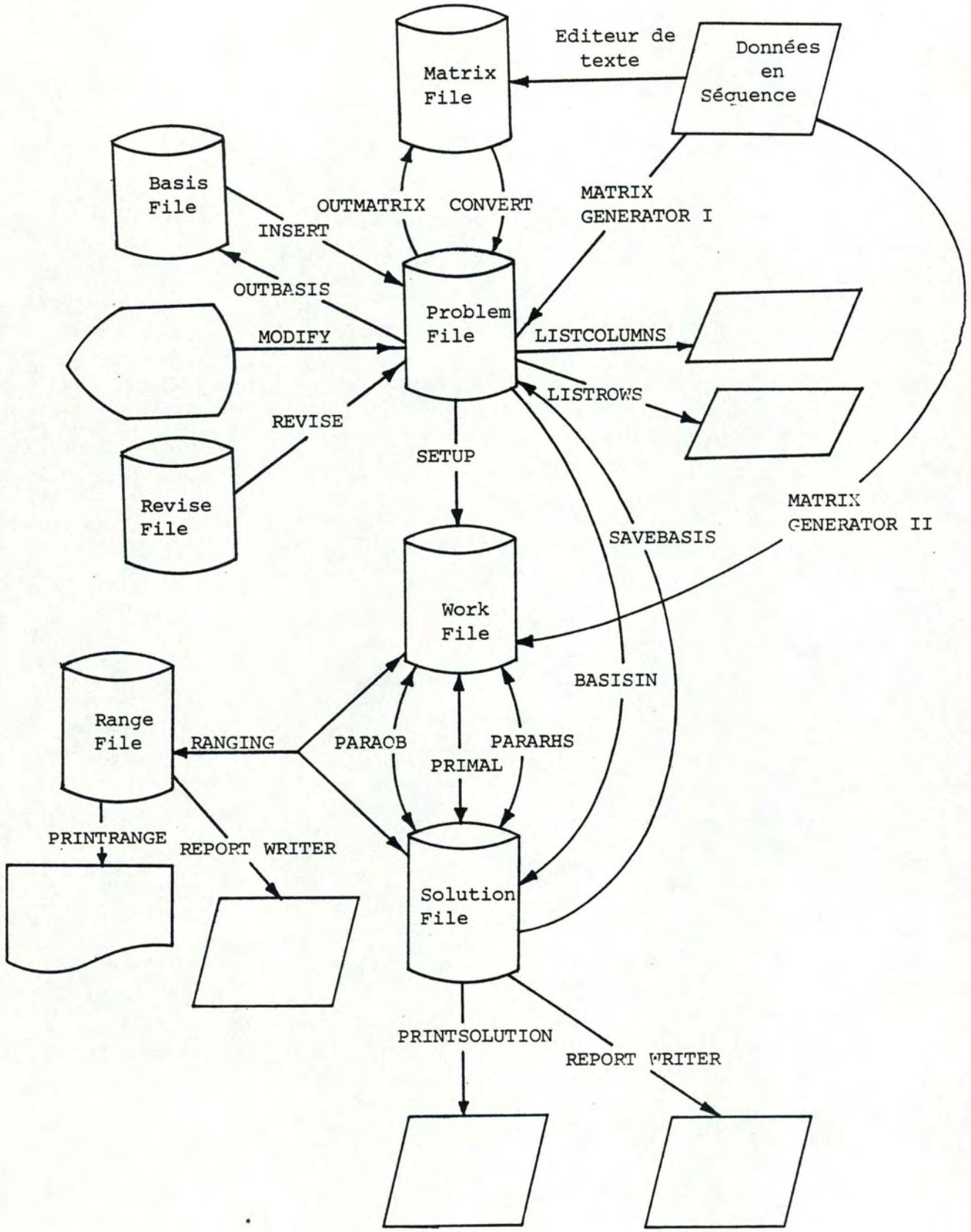

Matrix File
Editeur de texte
Données en Séquence
Basis File
OUTMATRIX
CONVERT
MATRIX GENERATOR I
INSERT
OUTBASIS
Problem File
MODIFY
LISTCOLUMNS
LISTROWS
REVISE
SETUP
Revise File
MATRIX GENERATOR II
SAVEBASIS
Work File
BASISIN
Range File
RANGING
PARAOB
PRIMAL
PARARHS
PRINTRANGE
REPORT WRITER
Solution File
PRINTSOLUTION
REPORT WRITER

Figure 4.1

Pour la résolution d'un problème de PLS, le logiciel LAMPS utilise des fichiers qui doivent être créés par l'utilisateur, d'autres fichiers sont générés par l'exécution de ses modules.

4.2.1. Fichiers créés par l'utlisateur

Les fichiers que l'utilisateur peut créer et qui seront utilisés comme fichiers d'entrée par LAMPS sont les suivants:

MATRIX FILE : contient la matrice du problème en format externe.

BASIS FILE : contient une base du problème en format externe.

REVISE FILE : contient les révisions qu'on désire réaliser à une matrice, en format externe.

4.2.2. Fichiers générés par LAMPS

PROBLEM FILE : contient la matrice et les bases d'un problème en format interne; ce format est approprié pour les modules de LAMPS concernant la manipulation des donnnées

WORK FILE : contient la base du problème dans un format approprié pour les modules algorithmiques de LAMPS, et plus précisément pour l'algoritme PRIMAL qui calcule la solution optimale. Ces fichiers sont effacés à la fin d'une session avec LAMPS, ou bien au moment où on désire réaliser une autre optimisation.

SOLUTION FILE : contient les solutions du problème en format interne.

RANGE FILE : contient les résultats de l'analyse de sensibilité (RANGING) en format interne.

4.2.3. Gestion des entrées - sorties

4.2.3.1. Entrées des données

La matrice des données du problème peut être créée dans le PROBLEM FILE de deux façons :

a) par lecture du MATRIX FILE avec le module CONVERT; dans ce cas, l'utilisateur écrit la matrice dans MATRIX FILE par l'intermédiaire d'un éditeur de texte en gardant la séquence exigée par Lamps, avec en plus un format très spécifique (les valeurs des items doivent commencer à partir d'une position bien déterminée dans la structure de l'article).

b) en utilisant la routine MATRIX GENERATOR I, qui crée directement la matrice, sans passer par le MATRIX FILE, à partir des données qui gardent la séquence exigée par LAMPS.

De même, la création de la matrice dans le WORK FILE peut être réalisée de deux façons :

a) à partir de la matrice stockée dans PROBLEM FILE, par l'application du module SETUP.

b) en utilisant la routine MATRIX GENERATOR II qui crée directement la matrice qui sera manipulée par PRIMAL, sans passer par MATRIX FILE et PROBLEM FILE, à partir des données qui gardent la séquence exigée par LAMPS. Evidemment, dans ce cas on ne pourra pas utiliser les modules de LAMPS concernant la manipulation des données de la matrice

4.2.3.2. Sortie des résultats

Les résultats de l'optimisation du problème sont stockés dans le SOLUTION FILE et peuvent sortir sur l'imprimante ou sur l'écran du terminal par l'utilisation du module PRINTSOLUTION. Il existe aussi la possiblité de lire les solutions à partir de SOLUTION FILE en utilisant la routine REPORT WRITER.

4.2.3.3. Changements à la matrice du problème

LAMPS permet de réaliser des changements à l'information concernant la matrice du problème, soit interactivement (avec MODIFY) si le nombre de modifications est petit, soit à partir du REVISE FILE (avec REVISE) si le nombre de modifications est considérable.

4.2.4. Algorithmes d'optimisation

La matrice du problème dans le WORK FILE a un format qui est adéquat pour les exigences des algorithmes de programmation linéaire simple implémentés par LAMPS. Ceci donne une bonne performance du point de vue du temps d'exécution. Malheureusement, aucun des modules de manipulation de données ne peut accéder à l'information contenue dans ce fichier.

4.2.5. Analyse de Post-optimisation

LAMPS permet de réaliser l'analyse de sensibilité d'une solution (RANGING) et la paramétrisation sur le vecteur des coefficients de la fonction objectif (PARAOBJ), ou sur le vecteur de termes indépendants (PARARHS).

4.3. Principal défaut du LAMPS

Bien que LAMPS soit un logiciel très rapide pour réaliser les calculs, le fait qu'on ne peut pas disposer de la matrice inverse de la base est un inconvénient très lourd. C'est en effet une information qui permet d'effectuer, d'une manière complète et efficace, l'analyse de post-optimisation.

4.4. Disponibilité actuelle des modules LAMPS comme sous-routines

Pour la réalisation du présent travail, une partie seulement des modules du Lamps qui peuvent être utilisés comme sous-routines, ont été testés et mis au point. C'était la situation jusqu'à la fin du mois de Janvier / 84, date du démarrage de la partie programmation du mémoire.

La configuration du LAMPS en mode batch, utilisable actuellement, est très semblable à celle montrée dans la figure 4.2. Cette configuration, sans être de loin la meilleure, permet l'optimisation d'un problème de PLS.

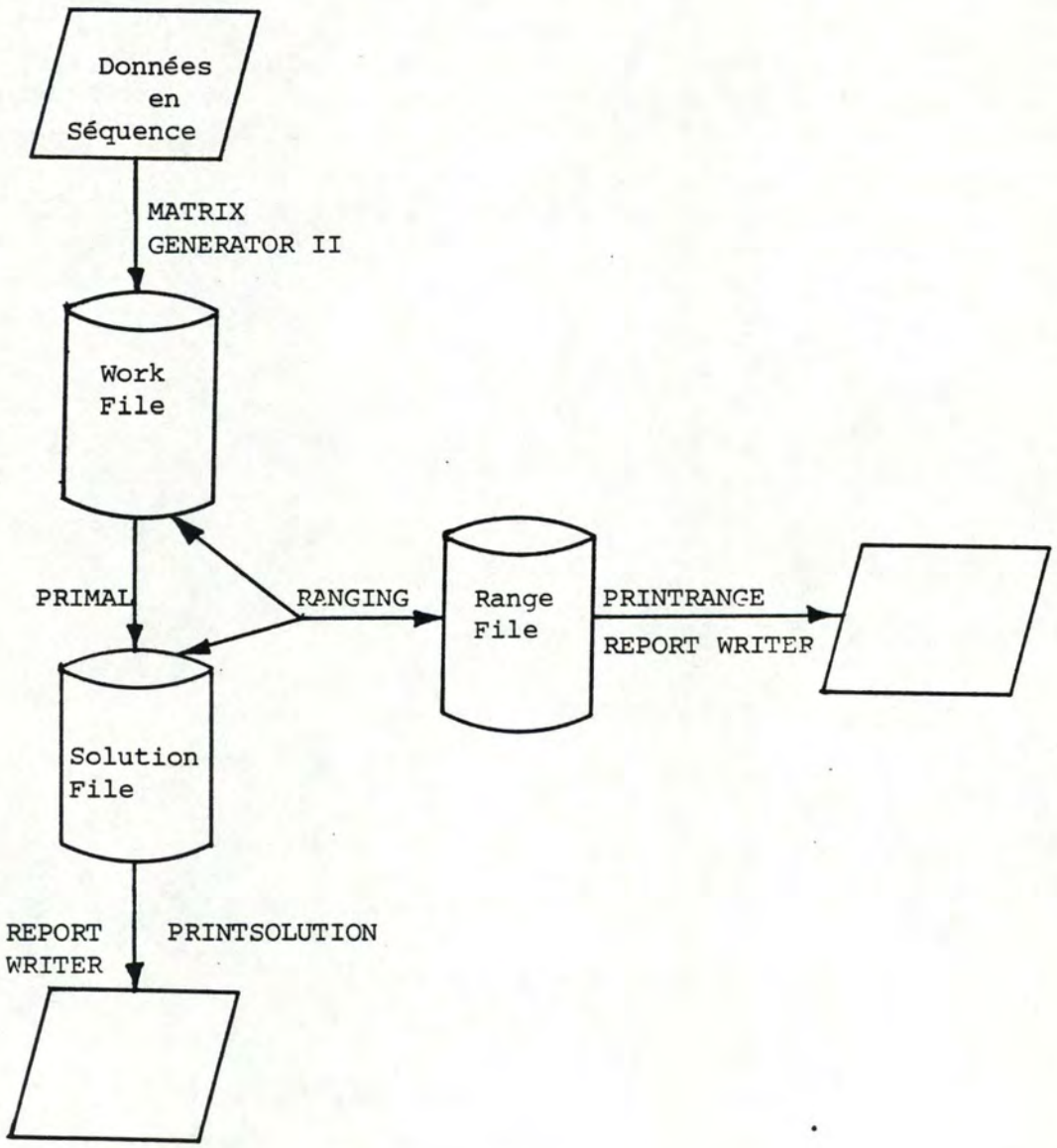


Figure 4.2

4.5. Restrictions sur les algorithmes de PLMO à cause des limitations actuelles du LAMPS en mode batch

Par le fait que la matrice n'existe pas dans le PROBLEM FILE, il faut la créer dans le WORK FILE, à partir des données de la matrice en séquence (avec la routine MATRIX GENERATOR II) chaque fois qu'on veut optimiser un problème. Cela signifie que pour optimiser chacune des fonctions objectif d'un problème de PLMO, il faut créer la matrice dans le WORK FILE, même si les contraintes du problème n'ont pas été changées.

Pour la même raison que celle indiquée ci-dessus, on ne peut pas réaliser le sauvetage d'une base du problème, et toute optimisation, sans exception doit commencer par la recherche d'une base initiale.

Les changements à apporter à la matrice du problème doivent être également réalisés sur les données en séquence, par des programmes externes au Lamps.

L'analyse de sensibilité réalisée par le module RANGING n'est pas suffisante pour la PLMO. De plus, comme Lamps ne fournit pas la matrice inverse de la base [cfr 4.3.], l'analyse de post-optimisation est pratiquement exclue pour les algorithmes de PLMO.

Chapitre 5:

IMPLEMENTATION DU SYSTEME DE PROGRAMMATION LINEAIRE MULTIOBJECTIFS

5.1. Introduction

Le logiciel a été conçu de telle façon que, même un utilisateur qui ne serait pas informaticienne et qui ne connaîtrait pas, à priori, les fondements et caractéristiques de la programmation linéaire multiobjectifs puisse l'utiliser facilement.

Le diagramme de flux (ou schéma) du système est représenté à la figure 5.1, et son architecture physique est reprise rapidement à la figure 5.2 (suivant la notation de Jackson[2]). Le système de PLMO comprend trois modules principaux exécutés séquentiellement chaque fois qu'on l'utilise. Ce sont l'introduction des données, les traitements (exécutions et/ou actualisations) et les clôtures.

Le module, d'exécution d'un problème de PLMO, utilise les routines de LAMPS pour résoudre les sous-problèmes de PLS générés par les méthodes de PLMO implémentées.

La structure du logiciel est modulaire, ce qui permet que certains modules soient appelés par plusieurs autres modules; l'insertion d'une nouvelle méthode dans le logiciel sera aisée pour la même raison.

Avant de décrire en détail chacun des trois modules, il faut signaler que la routine qui réalise la saisie des données est écrite en langage Cobol (toutes les autres routines du logiciel sont écrites en langage Fortran, imposé par le fait même que Lamps est écrit dans ce langage). Sur DEC-20, il n'est pas possible d'appeler une sous-routine Cobol à partir d'un programme Fortran; d'où la nécessité de passer au programme Fortran les données sorties du programme Cobol, stockées sur des fichiers.

La description des modules comprend d'abord le diagramme de flux, la description fonctionnelle, les entrées/sorties; cette information permet au lecteur d'avoir une idée générale sur le logiciel, mais suffisante pour l'utiliser. La structure physique et la description detaillée permettront aux lecteurs qui le

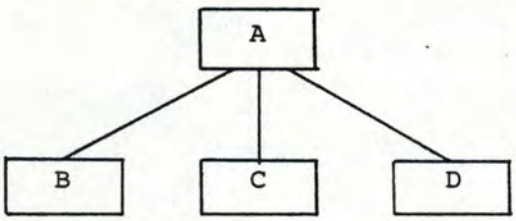
désirent, d'approfondir dans les détails l'architecture du logiciel.

---

(2) On peut représenter l'ensemble de traitements comme une structure arborescente en utilisant 4 types d'élements:

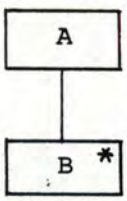
- La composante élementaire (feuille de l'arborescence).

- La séquence , qui est composée de deux ou plusieurs élèments qui se produisent chacun une fois, dans un ordre prè-defini. La structure est représentée comme:



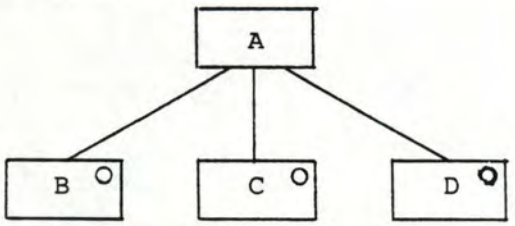
Le traitement A consiste à realiser les sous-traitements B, puis C, puis D.

- L'iteration , qui est composée d'un élèment qui se produit zero ou plusieurs fois consécutivement. La structure est représentée comme:



L'étoile dans le cadre de B indique les ocurrences multiples de B dans A. Il faut noter que A est l'iteration et que la multiplicité des occurrences de B est un attribut de A. La terminaison est déterminée par un test de condition.

- La sélection , est constituée de deux ou plusieurs élèments dont un seul se produit. La structure est représentée comme:



Le rond dans le cadre de B, C, et D indique que A est une sélection parmi ces trois traitements. La sélection est opérée sur base d'une condition.

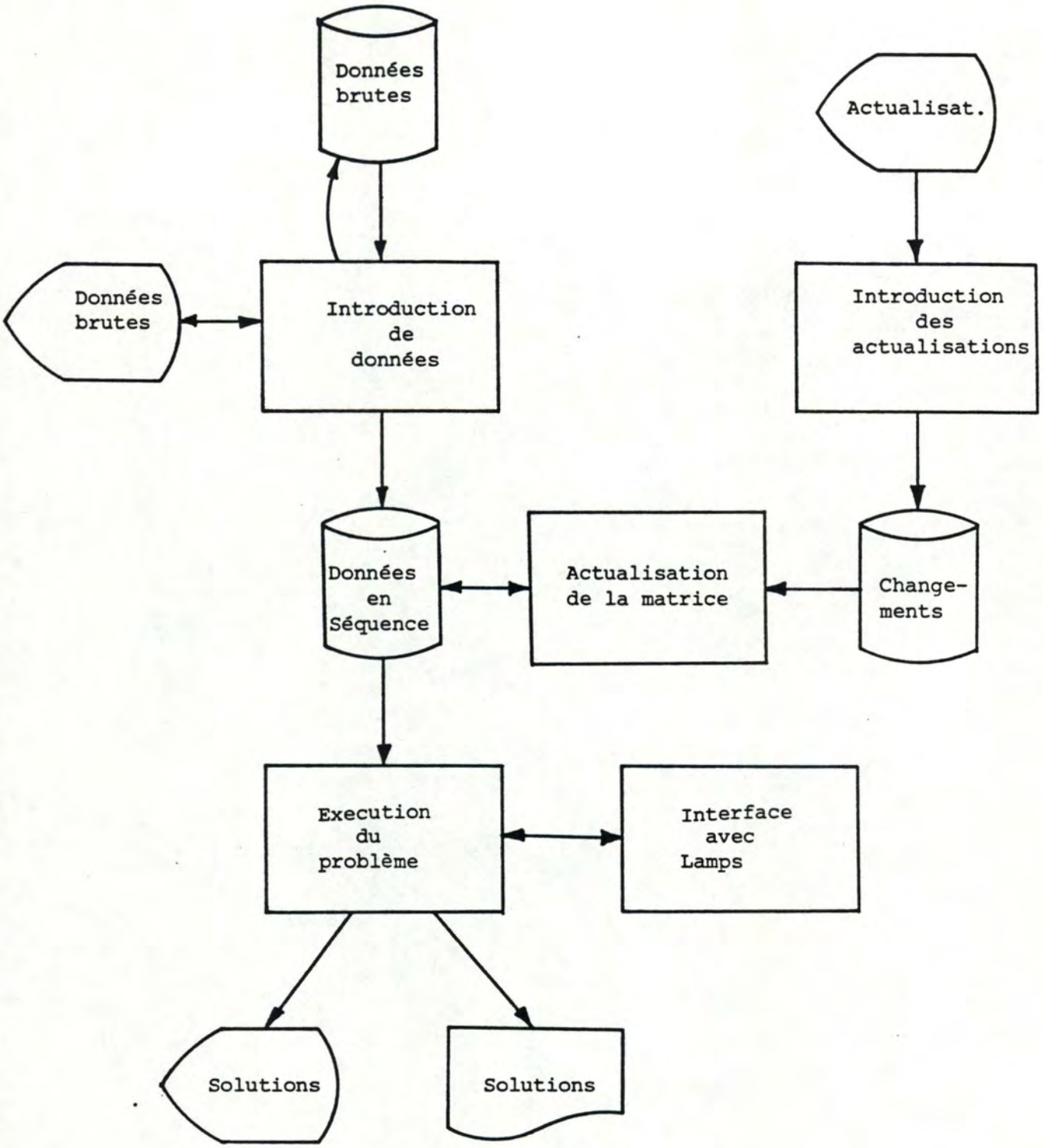


Figure 5.1. Diagramme de flux du Système de PLMO.

Notons que le seul fichier qui peut être créé et/ou actualisé par l'utilisateur est le fichier de données brutes (en utilisant un éditeur de texte). Tous les autres fichiers sont créés et/ou actualisés par le Logiciel.

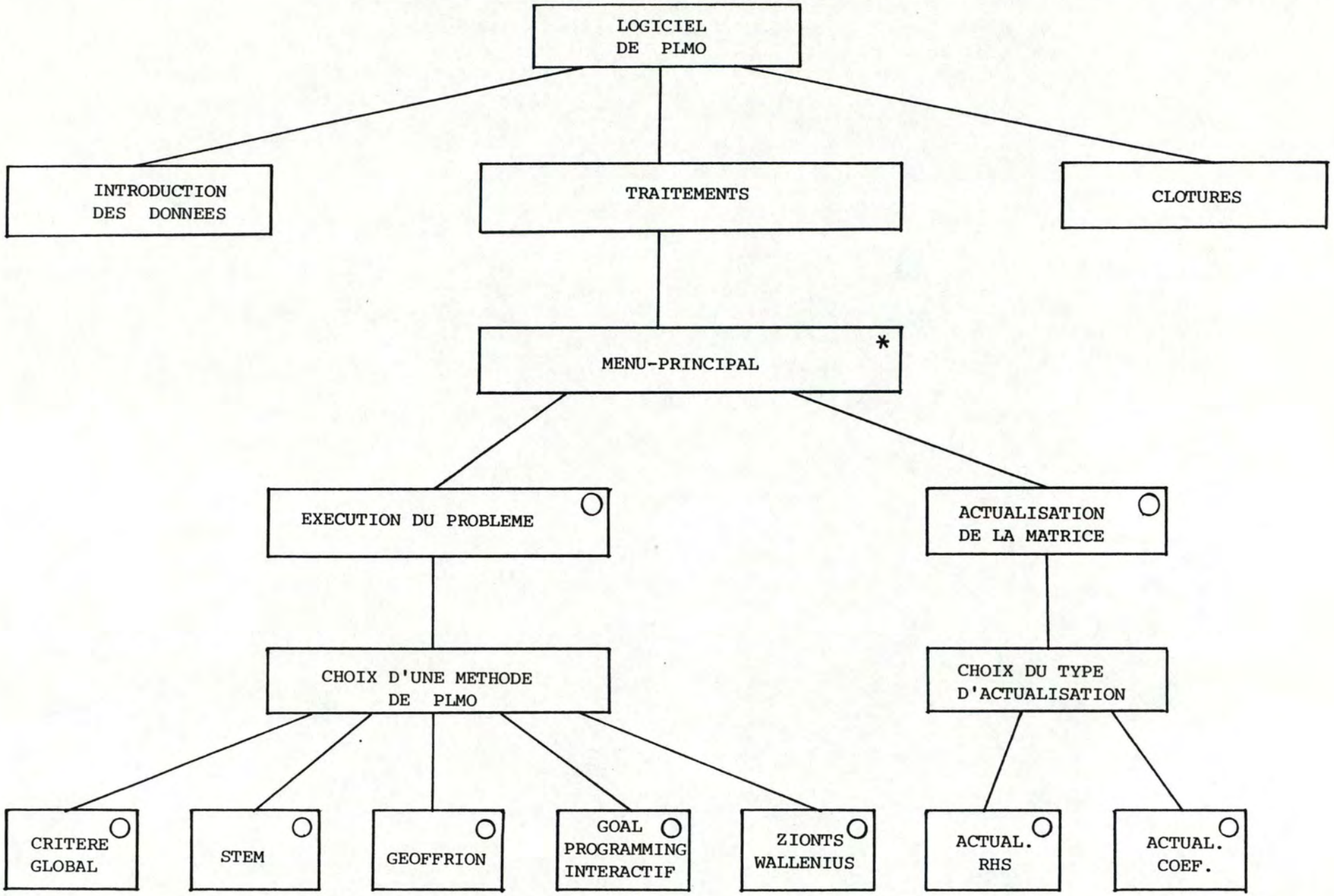


Figure 5.2. Architecture principale du Système de PLMO.

5.2. Module d'introduction des données

5.2.1. Objectif

Le but est de simplifier et de réduire considérablement la tâche d'introduction des données de la matrice du problème. Rappelons que celle-ci permet à l'utilisateur de fournir les données sous la "forme courante" d'une équation (cfr 6.3). Le module produit alors, en sortie, les données dans une séquence qui satisfait les exigences de la routine Matrix Generator II de Lamps [cfr] qui sont les suivantes:

	ITEM	TYPE	Longueur de l'item
1)	- Nom de la matrice	alphanumérique	8
	- commentaire	literal	72
2)	Pour chaque ligne de la matrice des données:		
	- Nom de la ligne	alphanumérique	12
	- Type de relation	alphanumérique	1
3)	Pour chaque terme non nul de la matrice des données:		
	- Nom de la colonne	alphanumérique	12
	- Nom de la ligne	alphanumérique	12
	- Valeur du terme	réel (double précision)	
4)	Pour chaque terme indépendant (RHS) des équations du problème:		
	- Nom du vecteur RHS	alphanumérique	12
	- Nom de la ligne	alphanumérique	12
	- Valeur du RHS	réel (double précision)	

Il faut remarquer dans 3), que Lamps réalise le rangement de la matrice par colonnes. Le module doit donc fournir les termes en respectant cette contrainte. Dans 2), le type de relation permet d'indiquer si la ligne représente une fonction objectif (N), une inégalité $\geq$ (G), une inégalite $\leq$ (L) ou, une égalite (E).

5.2.2. Diagramme de flux

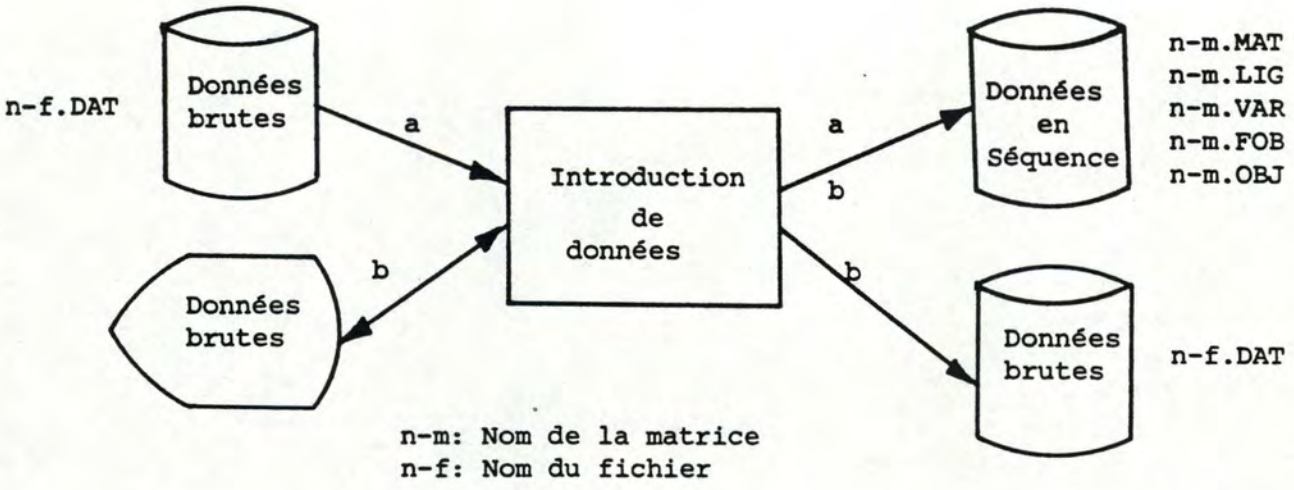


Figure 5.3

5.2.3. Description fonctionnelle

Réalise la saisie des données brutes du problème, écrites sous la forme courante des équations, et crée des fichiers contenant les données du problème en séquence qui seront manipulés par les méthodes de PLMO et par les routines de Lamps.

5.2.4. Entrée

Les équations (données brutes) du problème sont saisies d'une des deux façons suivantes:

a) à partir d'un fichier qui a été créé préalablement en utilisant un éditeur de texte, ou,
b) interactivement lors de l'exécution du module.

5.2.5. Sortie

- Des fichiers contenant les données de la matrice du problème en séquence.
- Eventuellement, pour le cas b) de l'entrée des données brutes, un fichier contenant les équations du problème.

5.2.6. Structure physique

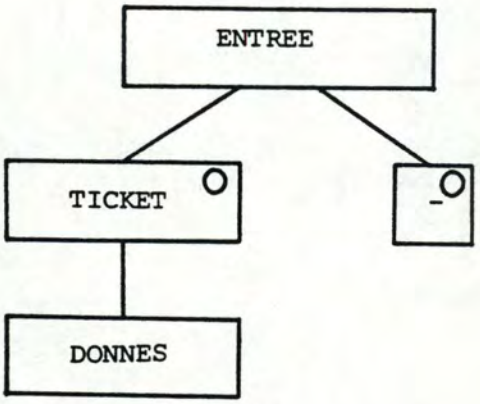


Figure 5.4

5.2.7. Description détaillée

Routine ENTREE

rôle: demander à l'utilisateur le nom de la matrice du problème; si les données en séquence pour cette matrice existent déjà, le contrôle d'exécution passe au module de traitements; dans le cas contraire, la routine crée le fichier

MATRIX.NAM = (nom-matrice).

paramètres: ifed

ifed= indique si l'entrée de données s'est bien achevée ("OK") ou s'il y a des erreurs ("NO").

Routine TICKET

rôle: demander à l'utilisateur le nom du fichier qui contient (ou qui contiendra) les équations du problème, et la facon dont il veut entrer les équations (par fichier ou interactivement).

paramètres:

la routine reçoit implicitement comme paramètre le nom de la matrice dans le fichier **MATRIX.NAM**

Routine DONNES

rôle: réaliser la transformation des données écrites sous

forme d'équations, en données qui ont un format et une séquence nécessaires pour être manipulées par les routines de calcul. S'il n'y a pas d'erreurs dans l'écriture des équations (dans le cas intéractif, la routine permet la correction d'erreurs, en nombre limité, pour chaque équation), les fichiers des données en séquence [cfr figure 5.3] sont créés suivant la structure suivante:

nom-matrice.MAT=(nom-matrice,choix,ifed)

nom-matrice.LIG=(nom-ligne,type-relation,valeur-THS)*

nom-matrice.VAR=(nom-ligne,nom-colonne,type-relation,valeur-terme)*

nom-matrice.OBJ=(nom-objectif,nom-colonne,type-relation,valeur-terme)*

nom-matrice,FOB=(nom-objectif,type-optimisation)*

L'information contenue dans le fichier nom-matrice.OBJ est redondante, mais il a été créé pour rendre plus efficace l'accès à cette information.

La structure du fichier des données brutes (nom-fichier.DAT) ainsi que les normes pour l'écriture des équations, sont reprises dans l'annexe et dans le chapitre suivant.

paramètres: nom-fichier,nom-matrice,choix

nom-fichier= nom du fichier contenant les équations du problème.

nom-matrice= nom de la matrice du problème utilisé pour assigner des noms à tous les fichiers générés par la routine. Ce nom est aussi utilisé par Lamps pour assigner des noms aux fichiers qu'il doit créer.

choix= manière d'introduire les données brutes (1: par fichier; 2: intéractivement).

5.3. Module de traitements

5.3.1. Objectif

Exécution du problème de PLMO par une des méthodes étudiées dans le chapitre III, en utilisant comme outil de base les routines de Lamps. Comme indiqué dans [4.5], actuellement on ne peut pas utiliser les routines de Lamps pour effectuer des changements à la matrice du problème; il faut donc développer dans le logiciel de PLMO les procéduresqui effectueront des actualisations.

Normalement, au cours de l'utilisation du logiciel pour exécuter un problème donné, l'utilisateur souhaitera sortir

la plus grande quantité d'information sur les solutions possibles au problème; il pourrait par exemple vouloir

- réexécuter le problème avec la même, ou une autre, méthode du logiciel (ce qui lui permettra de comparer l'efficacité et/ou la facilité d'utilisation de chaque méthode).

- apporter des petits changements à la matrice du problème et réexécuter celui-ci.

Pour que l'utilisateur puisse avoir une bonne idée de la démarche de la méthode qu'il a choisie, on lui donnera la possibilité de regarder le déroulement d'un problème-exemple.

5.3.2. Décomposition du module

Vu la nature différente des traitements à effectuer dans le module, celui-ci est divisé en deux sous-modules: le module d'exécution, et, le module d'actualisation. La routine GERANT permet à l'utilisateur de sélectionner un des deux sous-modules .

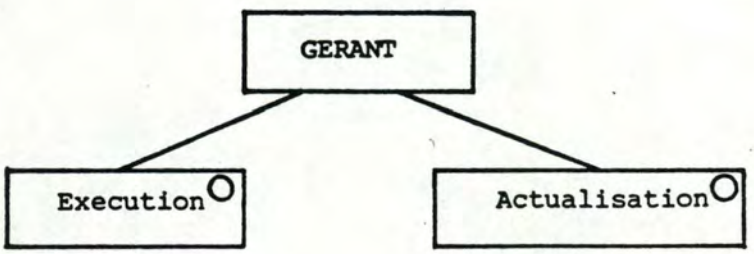


Figure 5.5

Le module de traitements comprend également le sous-module chargé de réaliser l'interface avec Lamps pour résoudre les sous-problèmes de PLS générés par les méthodes de PLMO.

5.3.3. Module d'exécution

5.3.3.1. Diagramme de flux

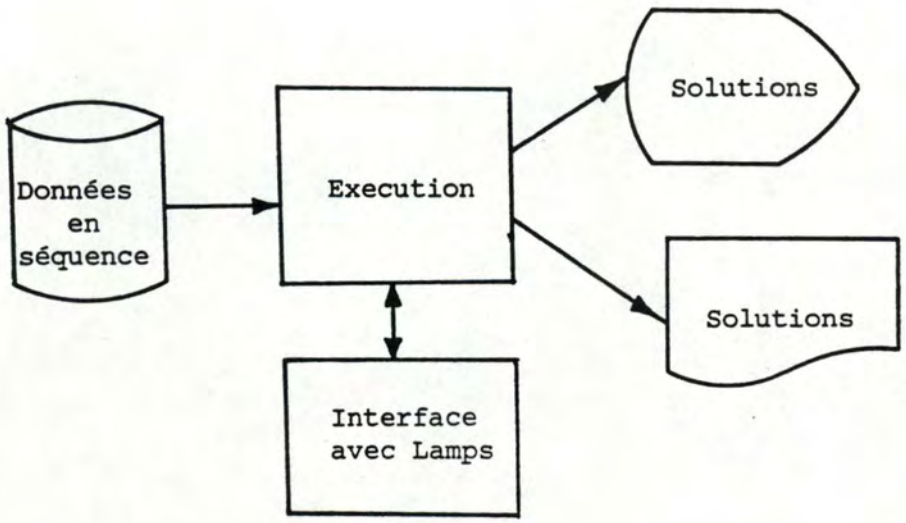


Figure 5.6

5.3.3.2. Description fonctionnelle

Exécute le problème avec la méthode de PLMO choisie par l'utilisateur. Le problème peut être réexécuté autant de fois que l'utilisateur le désire en utilisant la même méthode ou une autre.

5.3.3.3. Entrée

Les fichiers contenant les données en séquence générées par le module d'introduction des données.

5.3.3.4. Sortie

Les résultats de chaque solution (efficace ou possible) trouvés lors de l'exécution du problème, sont affichés à l'écran, et si l'utilisateur le désire, ils sont sortis sur l'imprimante.

5.3.3.5. Structure physique

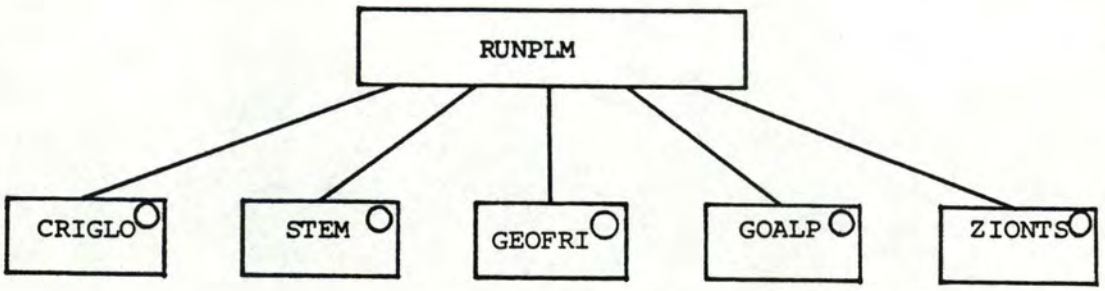


Figure 5.7

5.3.3.6. Description détaillée

Routine RUNPLM

rôle: présente le menu des méthodes du logiciel de PLMO et demande à l'utilisateur son choix.

Afin d'éviter une répétition, la description des paramètres ainsi que celle des sous-routines ne seront plus reprises à la suite des présentation des routines décrivant les méthodes du logiciel de PLMO. En effet, ces renseignements sont repris dans l'annexe.

Routine STEM

rôle: exécution du problème selon l'algoritme décrit dans [3.5.4].

structure physique: [cfr figure 5.8].

Routine GEOFRI

rôle: exécution du problème selon l'algorithme décrit dans [3.3.6]. La solution initiale (pas #1) est déterminée en utilisant les procédures qui calculent la première solution dans STEM.

structure physique: [cfr figure 5.9].

Routine GOALP

rôle: exécution du problème selon l'algorithme décrit dans [3.4.3]. La solution initiale (pas #2) est calculée par la résolution du problème (3.18) avec des poids wi = 1 , i=1,2,....,k.

structure physique: [cfr figure 5.10].

Routine ZIONTS

rôle: exécution du problème selon l'algorithme décrit dans [3.6.6]. Par le fait qu'on ne dispose pas avec Lamps de la matrice inverse de la base, la routine effectue la résolution d'un programme linéaire pour chaque variable hors-base (pas #3 de l'algorithme), ce qui rend la méthode encore plus inefficace.

structure physique: [cfr figure 5.12].

Routine CRIGLO

rôle: exécution du problème selon l'algorithme décrit dans [3.2.1].

structure physique:

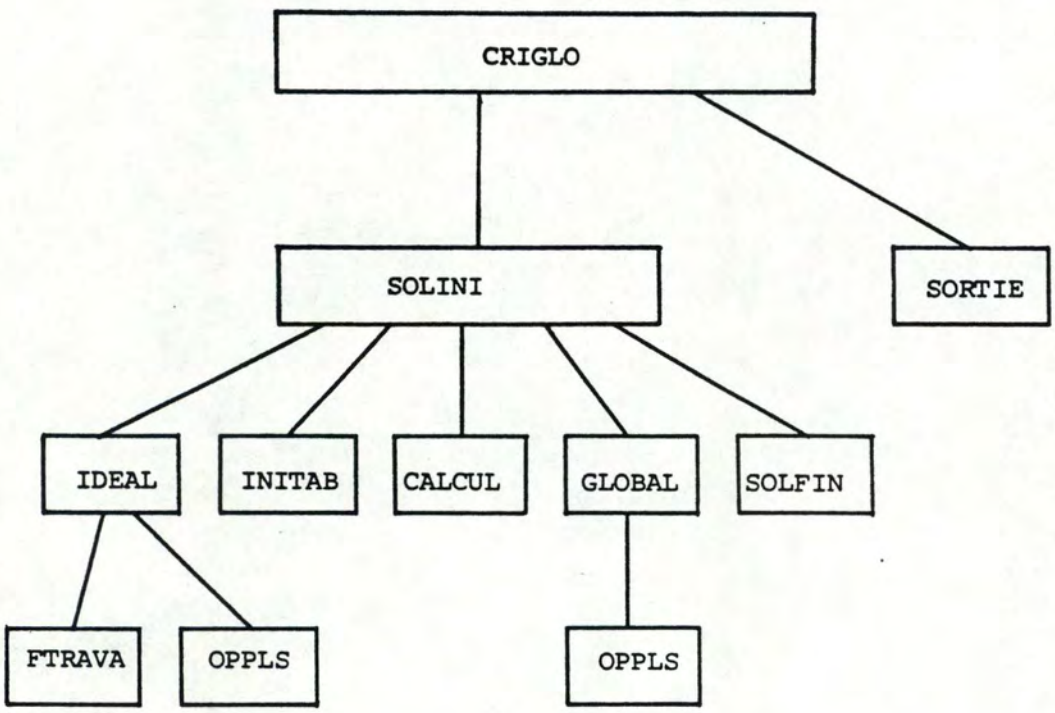


Figure 5.11

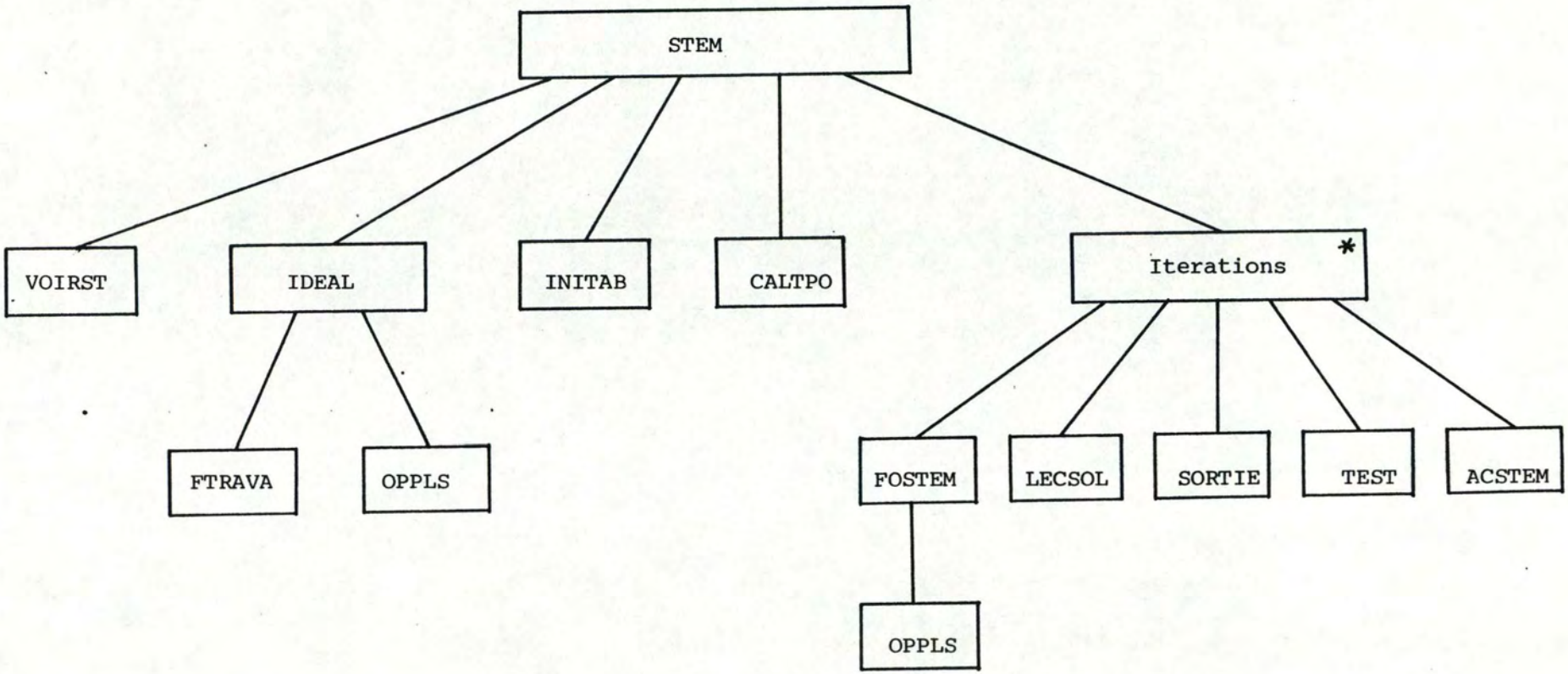

STEM
VOIRST
IDEAL
INITAB
CALTPO
Iterations
*
FTRAVA
OPPLS
FOSTEM
LECSOL
SORTIE
TEST
ACSTEM
OPPLS

Figure 5.8

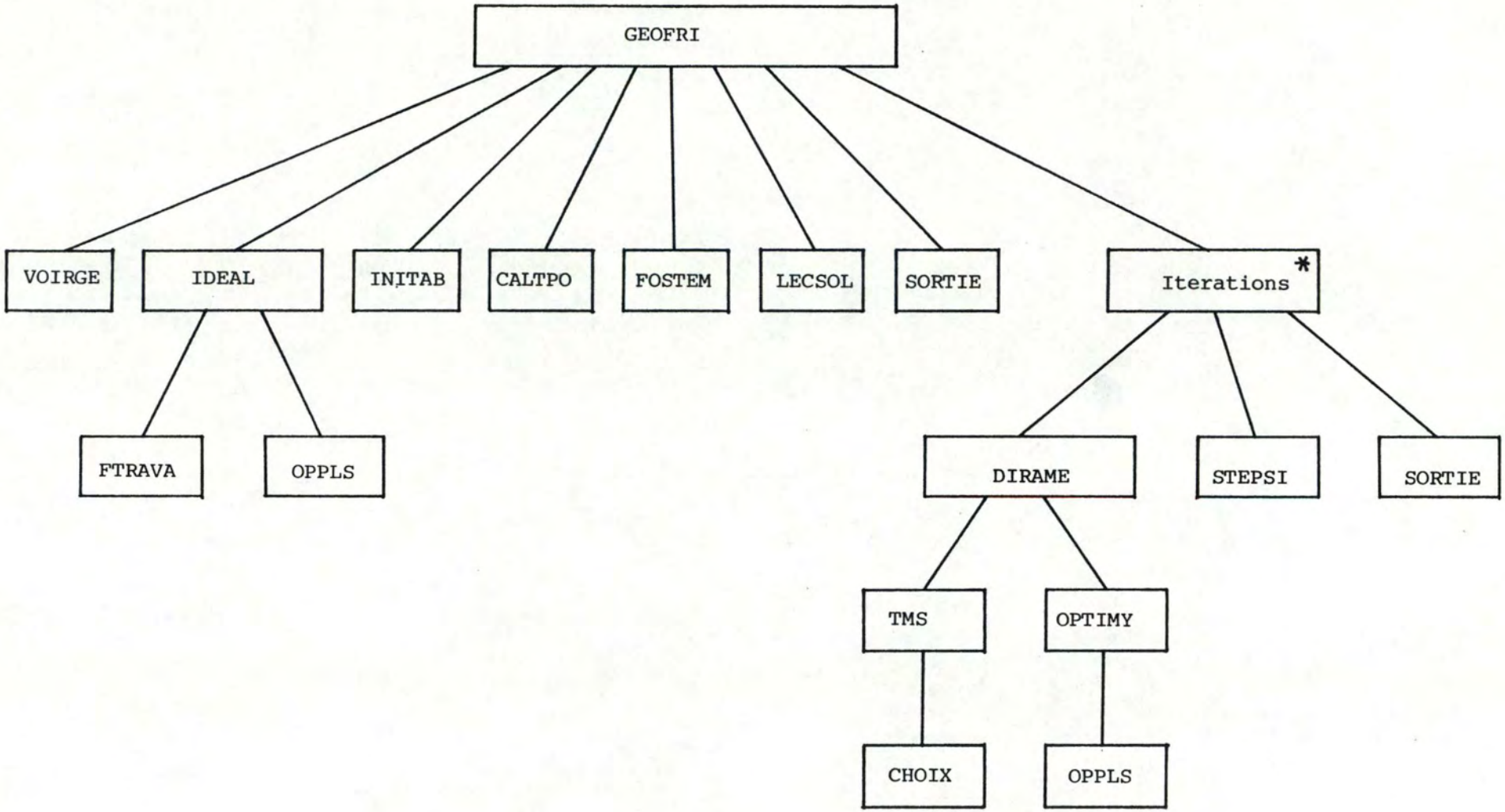

GEOFRI
VOIRGE
IDEAL
INITAB
CALTPO
FOSTEM
LECSOL
SORTIE
Iterations *
FTRAVA
OPPLS
DIRAME
STEPSI
SORTIE
TMS
OPTIMY
CHOIX
OPPLS

Figure 5.9

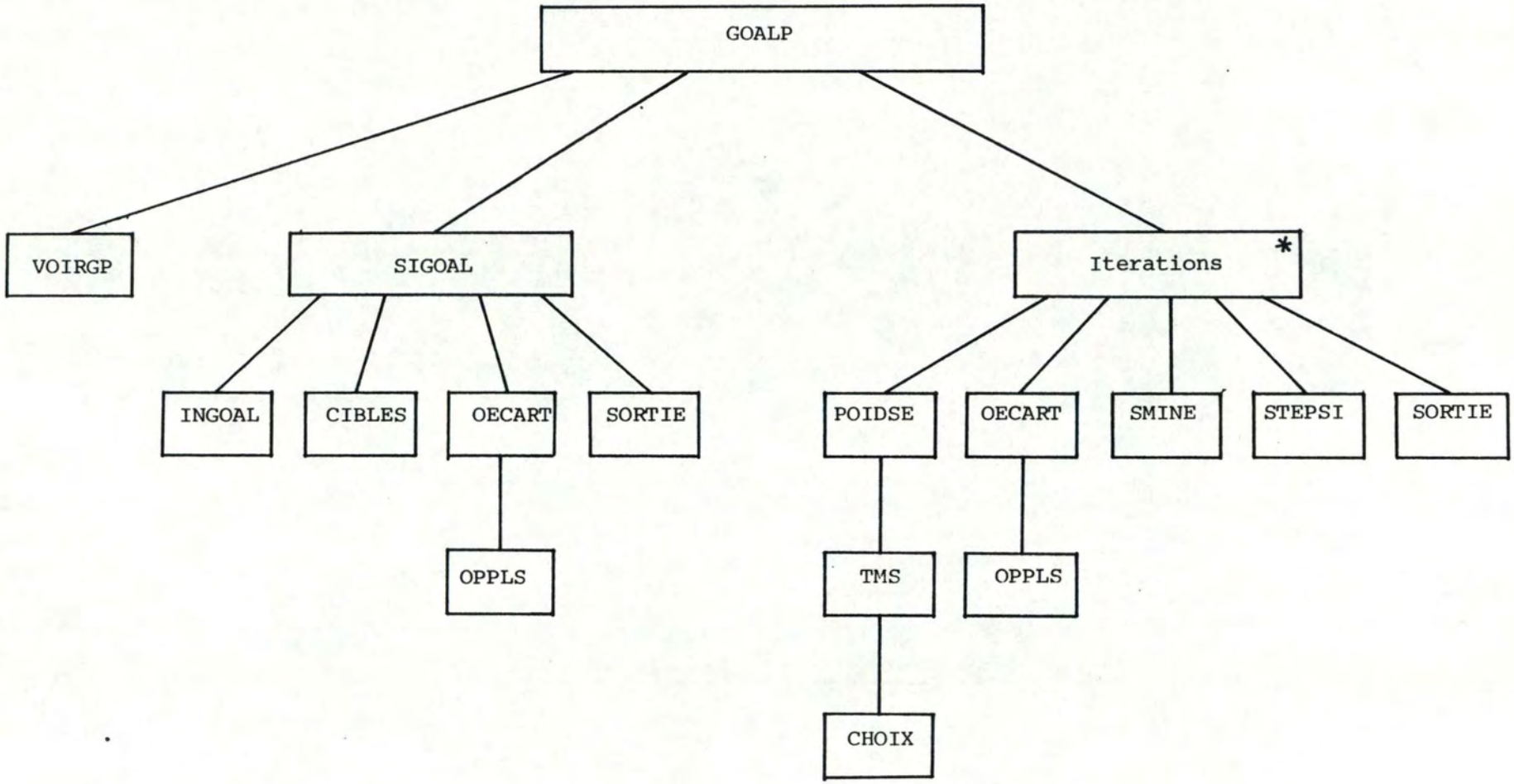

GOALP
VOIRGP
SIGOAL
Iterations
*
INGOAL
CIBLES
OECART
SORTIE
POIDSE
OECART
SMINE
STEPSI
SORTIE
OPPLS
TMS
OPPLS
CHOIX

Figure 5.10

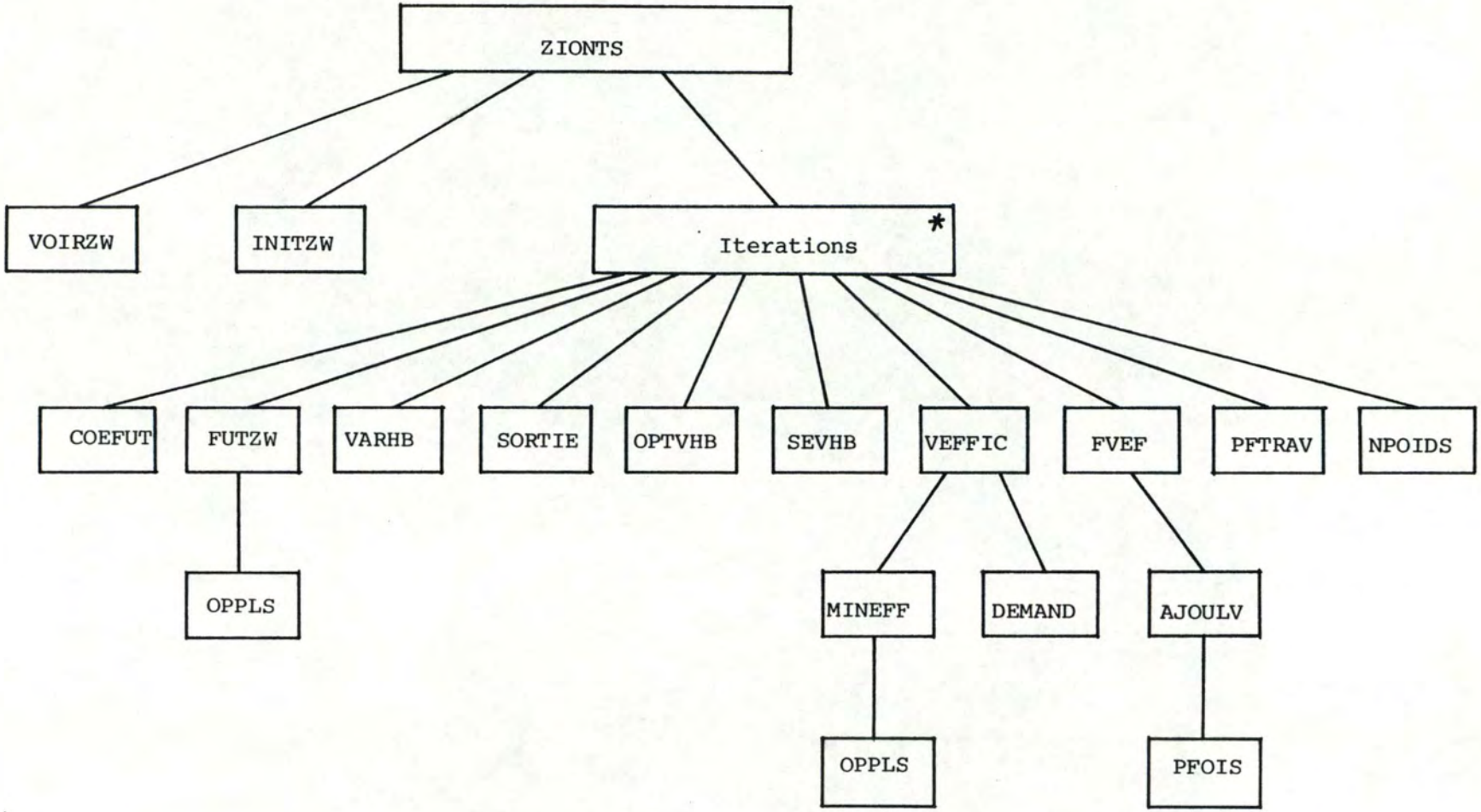

ZIONTS
VOIRZW
INITZW
Iterations
*
COEFUT
FUTZW
VARHB
SORTIE
OPTVHB
SEVHB
VEFFIC
FVEF
PFTRAV
NPOIDS
OPPLS
MINEFF
DEMAND
AJOULV
OPPLS
PFOIS

Figure 5.12

5.3.4. Module d'interface avec Lamps

5.3.4.1. Diagramme de flux

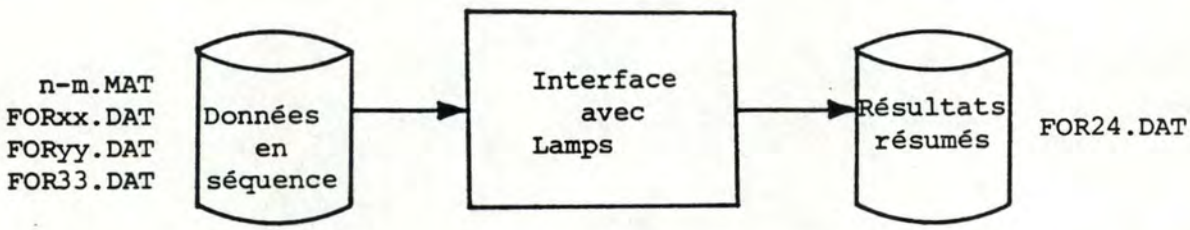


Figure 5.13

5.3.4.2. Description fonctionnelle

Résoud un problème de PLS, demandée par une des méthodes de PLMO, en utilisant les routines de Lamps disponibles actuellement.

5.3.4.3. Enrée

Des fichiers auxiliaires contenant les données de la matrice en séquence:

FORxx.DAT=(nom-ligne,type-relation,valeur-RHS)*

FORyy.DAT=(nom-ligne,nom-colonne,type-relation,valeur-terme)*

FOR33.DAT=(nom-objectif,type-optimisation)*

nom-matrice.MAT=(nom-matrice,choix,ifed)

5.3.4.4. Sortie

Un fichier contenant les résultats de la resolution du problème de PLS (si celui-ci possède un solution optimale), d'une manière résumée:

FOR24.DAT=(nom-objectif,nom-variable,valeur)*

Dans l'article qui contient la valeur de l'objectif, le nom de la variable est le nom de l'objectif.

5.3.4.5. Structure physique

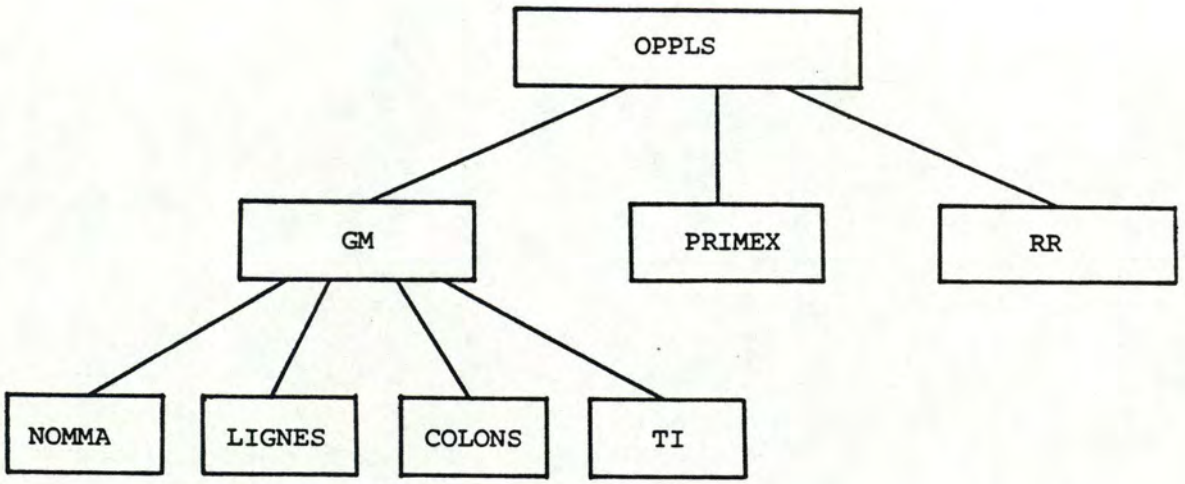


Figure 5.14

5.3.4.6. Description detaillée

Routine OPPLS

rôle:

- Création de la matrice du problème dans le fichier Work File en utilisant la routine Matrix Generator II (GM).
  On réserve d'abord de la place en mémoire secondaire (NOMMA) pour les fichiers qui seront générés par Lamps (plus précisement les fichiers: Work File = nom-matrice.WRK, et Solution File = nom-matrice.SOL).
  Ensuite, on introduit dans le Work File les informations concernant les lignes de la matrice (LIGNES), les termes de la matrice (COLONS), et les RHS (TI).

- Exécution du problème stocké dans le Work File, en utilisant la routine Primal de Lamps (PRIMEX).

- Lecture des résultats de l'optimisation à partir du fichier Solution File; ils sont stockés dans le fichier de sortie FOR24.DAT (s'il existe une solution optimale), par la routine Report Writer de Lamps (RR).

paramètres: nomfm,inf,nolig,nocol,nfl,nfv,iecart

nomfm= nom du fichier qui contient le nom de la matrice du problème. Ce nom est utilisé afin de réserver de la place sur disque pour les fichiers qui générés par Lamps.

inf= indique si le problème de PLS possède une solution optimale (valeur 0) ou non (valeur 1).

nolig= nombre de lignes de la matrice du problème.

nocol= nombre de colonnes de la matrice du problème.

nfl= numéro du fichier contenant les informations sur les lignes de la matrice (et compris les valeurs des RHS).

nfv= numéro du fichier contenant les informations sur les termes de la matrice.

iecart= indique si le module doit fournir les valeurs des variables d'écart dans la solution optimale.

5.3.5. Module d'actualisations

5.3.5.1. Objectif

Offrir à l'utilisateur le moyen d'apporter quelques changements aux valeurs des RHS, ou aux coefficients de la matrice, pour qu'il puisse poursuivre l'analyse des solutions du problème en réexécutant le programme.

5.3.5.2. Diagramme de flux

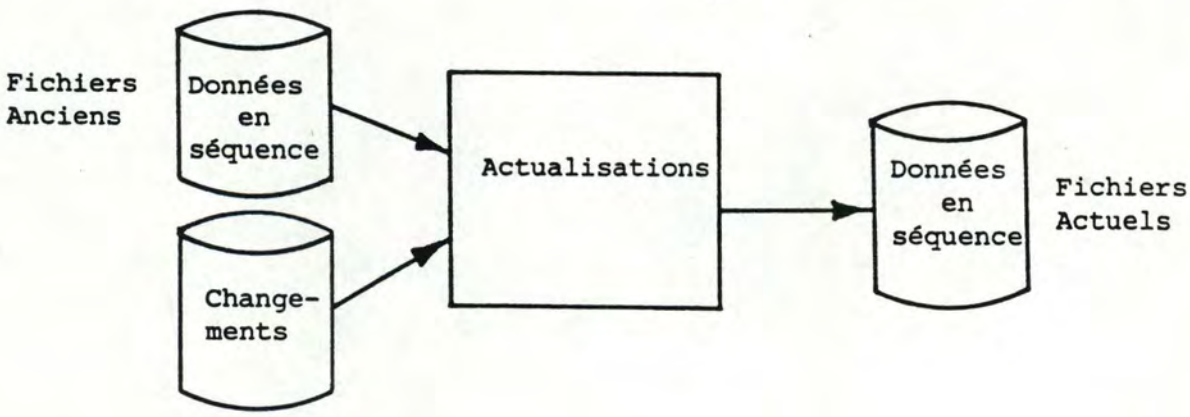


Figure 5.15

5.3.5.3. Description fonctionnelle

Actualisation des données en séquence, en utilisant la technique de fusion de fichiers.

5.3.5.4. Entrée

- Des fichiers contenant les anciennes données de la matrice en séquence.
- Le fichier de changements est crée interactivement avec l'utilisateur.

5.3.5.5. Sortie

Les fichiers actualisés contenant les données de la matrice en séquence.

5.3.5.6. Structure physique

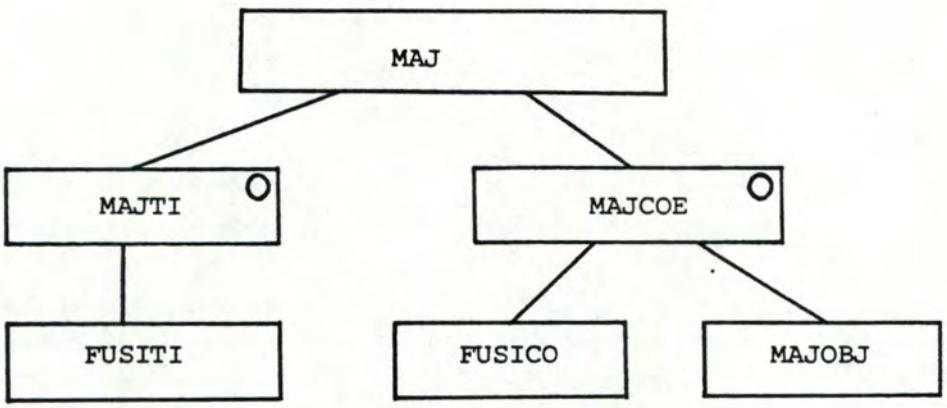


Figure 5.16

5.3.5.7. Description detaillée

Routine MAJ

rôle:

- Sélectionner le type d'actualisation à réaliser (termes indépendants ou coefficients) sur la matrice.
- Créer, interactivement avec l'utilisateur, le fichier contenant les changements à effectuer, soit aux termes indépendants (MAJTI) ou aux coefficients (MAJCOE).
- Effectuer les actualisations sur les données en séquence utilisant la technique de fusion

de fichiers, pour les termes indépendants (FUSITI) ou pour les coefficients (FUSICO). Pour le deuxième cas, il faudra peut être effectuer les actualisations sur les coefficients des fonctions objectifs (MAJOBJ).

Remarque: Une fois qu'on a effectué des changements dans les données en séquence de la matrice du problème, les équations du problème (données brutes) ne seront plus valables puisque elles ont conservé les données de départ. Dès lors, si l'utilisateur désire réaliser des actualisations comme ajout/suppression d'une ou plusieurs équations, ou d'une ou plusieurs variables, il est contraint à se servir d'un éditeur de texte.

5.4. Module Clôtures

5.4.1. Diagramme de flux

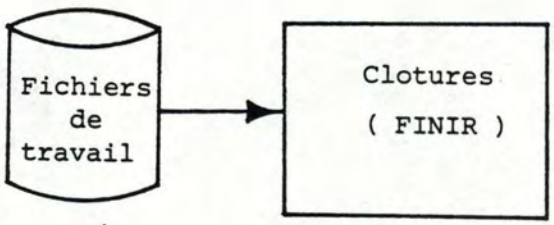


Figure 5.17

5.4.2. Description fonctionnelle

Effacer tous les fichiers auxiliaires (de travail) qui ont été créés par n'importe quelle méthode de PLMO lors de l'exécution du module de traitements

5.4.3. Entrée

Fichiers de travail créés par les méthodes de PLMO.

Chapitre 6: MANUEL DE L'UTILISATEUR ET EXEMPLE D'UTILISATION

6.1. Conventions concernant la lecture de ce chapitre

<R> = Pousser sur la touche RETURN du terminale.

□ = Position du curseur à l'ecran.

ϕ = Suite d'un ou plusieurs caractères blancs.

[ ] = L'expression encadrée dans ces symboles est facultative.

{ } = On doit choisir un et un seul élement parmi la liste d'expressions encadrée dans ces symboles.

... = Ce qui précède peut être répété un certain nombre de fois.

6.2. Exécution du programme de PLMO

- Si on dispose de la version exécutable du programme PLMO et du programme TICKET (PLMO.EXE et TICKET.EXE respectivement), alors il suffit de taper la commande

  @ RUN PLMO <R>

- Sinon, il faut passer par les commandes suivantes

  @ LOAD TICKET.CBL,DONNES.CBL <R>
  @ SAVE TICKET <R>
  @ COMPILE OPPLS.FOR <R>
  @ COMPILE PLMO.FOR <R>
  @ LINK <R>
  * @PLMO <R> (N.B. ici, @ c'est le caractère et non CTRL-C)
  @ RUN PLMO <R>

- Le programme demande à l'utilisateur le nom qu'il veut assigner à la matrice du problème à résoudre.

  TAPEZ LE NOM DE LA MATRICE DU PROBLEME
  (Maximun 6 caractères majuscules) :□

  Puisque le nom de la matrice est un item alphanumérique (formé de lettres et/ou de digits uniquement), la contrainte "caractères majuscules" ne porte bien sûr que sur les lettres. Si on tape plus de 6 caractères, le système ne

prend en compte que les 6 premiers.

- Si les fichiers contenant les données en séquence pour la matrice dont le nom vient d'être donné par l'utilisateur, existent déjà, le programme présente le menu principal (de traitements).
  Sinon, on demande à l'utilisateur le nom du fichier qui contient les données brutes (équations) du problème à résoudre (s'il a été déjà créé, en utilisant l'éditeur de texte ou par le système PLMO exécuté auparavant), ou qui les contiendra lors de l'exécution du logiciel.

  TAPEZ LE NOM DU FICHIER QUI CONTIENT (OU QUI CONTIENDRA)
  LES EQUATIONS DU PROBLEME
  (Maximum 6 caractères majuscules) :□

  Conseil :donner un nom différent à la matrice et au fichier afin de faciliter la tâche d'effacement de fichiers en utilisant la commande DELETE.

- Demande la façon dont l'utilisateur veut introduire les données brutes du problème

  MANIERE D'INTRODUIRE LES DONNEES

  PAR FICHIER ---> Taper 1

  PAR ECRAN -----> Taper 2

  Votre choix: □

  La première option (1) est valide seulement si le fichier contenant les données brutes existe déjà.

6.3. Spécifications et normes pour écrire les équations

1. Une équation est une chaîne de caractères terminée par un point virgule (";").

   Cette chaîne de caractères se compose de:
   a) Un nom d'équation,
   b) Une suite alternée de termes (coefficient et nom de variable) et d'opérateurs arithmétiques "+" ou "-",
   c) - Un signe d'égalité ou d'inégalité suivi du terme indépendant, pour les contraintes,
      - Le signe "$" suivi d'un indicateur du type d'optimisation ("max" ou "min") pour les fonctions objectifs.

2. - Les signes "≥" et "≤" sont représentés par ">" et "<" respectivement
   - Tous les coefficients, même les entiers, doivent contenir explicitement le point décimal.

- Les coefficients unitaires doivent doivent être explicites (1.).
- La longueur maximale d'un nom de variable ou nom d'équation est de 12 caractères.

Suivant la notation Cobol, le format pour l'écriture d'une équation serait:

nom-équation ␣ { { + / - } [␣] coefficient ␣ nom-variable ␣ } ...

{ $ / > / < / = } { ␣ { MAX / MIN } / [␣] { [+] / - } [␣] coefficient } ␣ ;

où: coefficient = partie-entière.[partie-décimale]

A titre d'exemple , voici un problème de PLMO comme il se pose:

max Profit = x1 + 3 x2 + 3 x3

max Pollution = 3 x1 + 2 x2 + 3

s.c. g1: 0.5 x1 + 0.25 x2 - 0.7 x3 ≤ 8

g2: - 0.2 x1 + 0.2 x2 + 0.5 x3 = 4.75

g3: x1 + 5 x2 ≥ 72

x1,x2,x3 ≥ 0

Il sera écrit de la façon suivante:

```
PROFIT
+ 1. X1 + 3. X2 + 3. X3 $ MAX ;
POLLUTION
+ 3. X1 + 2. X2 + 1. X3 $ MIN ;
G1
+ 0.5 X1 + 0.25 X2 - 0.7 X3 < 8.0 ;
G2
- 0.2 X1 + 0.2 X2 + 0.5 X3 = 4.75 ;
G3
+ 1. X1 + 5. X2 > 72. ;
```

6.4. Menu principal (traitements)

Dès le début de l'exécution, le programme affiche le menu principal:

UTILISATION DU LOGICIEL DE PROGRAMMATION LINEAIRE MULTI-OBJECTIFS

POUR RESOUDRE LE PROBLEME ----------> Taper 1

POUR ACTUALISER LA MATRICE DE DONNEES ---> Taper 2

POUR FINALISER ----------> Taper 3

Votre choix: ☐

Si on choisit l'option 1 ou 2, on retourne à ce menu principal dès que l'exécution est terminée.

6.5. Menu d'exécution

METHODES POUR RESOUDRE UN PROBLEME DE PROGRAMMATION LINEAIRE MULTIOBJECTIFS

1: Critère Global
2: STEM
3: Zionts et Wallenius
4: Geoffrion
5: Goal Programming (Si vous disposez des valeurs cibles pour les objectifs)

Votre choix: ☐

6.6. Menu d'actualisation

ACTUALISATIONS

Tapez 1: si vous voulez changer la valeur d'un terme independant

Tapez 2: si vous voulez changer la valeur d'un coefficient

Votre choix: ☐

6.7. Effacement de fichiers

L'exécution terminée, si on ne désire pas garder les fichiers contenant les données en séquence de la matrice du problème, on peut toutes les effacer en donnant la commande

@ DELETE nom-matrice.*.* <R>

Au même temps, les fichiers nom-matrice.WRK et nom-matrice.SOL crées par Lamps, seront effacés.

6.8. Exemple d'utilisation

Afin d'illustrer l'utilisation du logiciel de PLMO, considèrons le problème de nourriture enoncé dans [14], pages 111-112.

Le problème de nutrition est de trouver les quantités de différents aliments qu'il faut consommer pour satisfaire certains besoins nutritifs à un coût minimum.

Le tableau ci dessous reprend les caractéristiques des différents produits alimentaires.

Le problème revient à trouver les quantités de nourriture répondant aux contraintes et objectifs suivants:

i) assurer l'apport minimun necessaire en vitamine A et en fer, ainsi que de garantir une consommation equilibrée en protéine et en énergie nutritive.
ii) minimiser la consommation en cholesterol et en hydrate de carbone.
iii) finalement, minimiser le coût de la nourriture.

Pour chaque produit, les maximums fixés sont les suivants:

6 pint (0.56 litres) de lait
1 pound (0.453 kg) de boeuf
1/4 de douzaine [dozen] d'oeufs
10 ounces (1 ounce = 28.5 gr) de pain
10 ounces de salade
4 pint de jus d'oranges

	Lait (pint)	Boeuf (pound)	Oeuf (dozen)	Pain (ounce)	Salade (ounce)	Jus d'orange (pint)	Récomendation journalière pour adulte
Vitamine A (I. U.)	720	107	7080	0	134	1000	5000
Energie nutritive (calories)	344	460	1040	75	17.4	240	2500
Cholèsterol (unité)	10	20	120	0	0	0	
Protéine (g)	18	151	78	2.5	0.2	4	63
Hydrate de carbone (g)	24	27	0	15	1.1	52	
Fer (mg)	0.2	10.1	13.2	0.75	0.15	1.2	12.5
Coût ($)	0.225	2.2	0.8	0.1	0.05	0.26	

Tableau 6.1

Le modèle mathématique est le suivant:

$$\min \quad z1(x) = 0.225\,x1 + 2.2\,x2 + 0.8\,x3 + 0.1\,x4 + 0.05\,x5 + 0.26\,x6$$

$$\min \quad z2(x) = 10\,x1 + 20\,x2 + 120\,x3$$

$$\min \quad z3(x) = 24\,x1 + 27\,x2 + 15\,x4 + 1.1\,x5 + 52\,x6$$

$$\text{s.c.} \quad g1(x) = 720\,x1 + 107\,x2 + 7080\,x3 + 134\,x5 + 1000\,x6 \geq 5000$$

$$g2(x) = 0.2\,x1 + 10.1\,2 + 13.2\,x3 + 0.75\,x4 + 0.15\,x5 + 1.2\,x6 \geq 12.5$$

$$g3(x) = 344\,x1 + 460\,x2 + 1040\,x3 + 75\,x4 + 17.4\,x5 + 240\,x6 = 2500$$

$$g4(x) = 18\,x1 + 151\,x2 + 78\,x3 + 2.5\,x4 + 0.25\,x5 + 4\,x6 = 63$$

$$g5(x) = x1 \leq 6.0$$

$$g6(x) = x2 \leq 1.0$$

$$g7(x) = x3 \leq 0.25$$

$$g8(x) = x4 \leq 10.0$$

$$g9(x) = x5 \leq 10.0$$

$$g10(x) = x6 \leq 4.0$$

$$xi \geq 0,\ i=1,2,\ldots,6$$

Dans les pages qui suivent, on présente les contenues des fichiers de données brutes, de données en séquence pour le problème.

L'exécution du problème pour les méthodes interactives est également reprise.

Fichier de donnees brutes (equations): NOURIP.DAT

```
Z1
0.225 X1 + 2.2 X2 + 0.8 X3 + 0.1 X4 + 0.05 X5 + 0.26 X6 s MIN :
Z2
10. X1 + 20. X2 + 120. X3 s MIN :
Z3
24. X1 + 27. X2 + 15. X4 + 1.1 X5 + 52. X6 s MIN :
G1
720. X1 + 107. X2 + 7080. X3 + 134. X5 + 1000. X6 > 5000. :
G2
0.2 X1 + 10.1 X2 + 13.2 X3 + 0.75 X4 + 0.15 X5 + 1.2 X6 > 12.5 :
G3
344. X1 + 460. X2 + 1040. X3 + 75. X4 + 17.4 X5 + 240. X6 > 2500. :
G4
18. X1 + 151. X2 + 78. X3 + 2.5 X4 + 0.2 X5 + 4. X6 > 63. :
G5
1. X1 < 6. :
G6
1. X2 < 1. :
G7
1. X3 < 0.25 :
G8
1. X4 < 10. :
G9
1. X5 < 10. :
G10
1. X6 < 4. :
```

Fichier contenant l'information sur les lignes et RHS:

TEST.LTG

```
Nom-ligne    ti  valeur-RHS

G1           G+0000050000000
G10          L+0000000400000
G2           G+0000001250000
G3           G+0000025000000
G4           G+0000006300000
G5           L+0000000600000
G6           L+0000000100000
G7           L+0000000025000
G8           L+0000001000000
G9           L+0000001000000
```

Fichier contenant l'information sur les termes de la matrice:

TEST.VAR

Nom-ligne	Nom-colonne	tr	valeur-terme
G1	X1		G+0000007200000
G2	X1		G+0000000020000
G3	X1		G+0000034400000
G4	X1		G+0000001800000
G5	X1		I+0000000100000
Z1	X1		N+0000000022500
Z2	X1		N+0000001000000
Z3	X1		N+0000002400000
G1	X2		G+0000010700000
G2	X2		G+0000001010000
G3	X2		G+0000046000000
G4	X2		G+0000015100000
G6	X2		I+0000000100000
Z1	X2		N+0000000220000
Z2	X2		N+0000002000000
Z3	X2		N+0000002700000
G1	X3		G+0000070800000
G2	X3		G+0000001320000
G3	X3		G+0000104000000
G4	X3		G+0000007800000
G7	X3		I+0000000100000
Z1	X3		N+0000000080000
Z2	X3		N+0000012000000
G2	X4		G+0000000075000
G3	X4		G+0000007500000
G4	X4		G+0000000250000
G8	X4		I+0000000100000
Z1	X4		N+0000000010000
Z3	X4		N+0000000150000
G1	X5		G+0000001340000
G2	X5		G+0000000001500
G3	X5		G+0000000174000
G4	X5		G+0000000002000
G9	X5		I+0000000100000
Z1	X5		N+0000000000500
Z3	X5		N+0000000011000
G1	X6		G+0000010000000
G10	X6		I+0000000100000
G2	X6		G+0000000012000
G3	X6		G+0000002400000
G4	X6		G+0000000040000
Z1	X6		N+0000000002600
Z3	X6		N+0000000520000

Fichier contenant l'information des termes

des fonctions objectifs: TEST.OBJ

Nom-objet. Nom-colonne tr valeur-terme

Z1	X1	N+00000000022500
Z1	X2	N+00000000220000
Z1	X3	N+00000000080000
Z1	X4	N+00000000010000
Z1	X5	N+00000000005000
Z1	X6	N+00000000026000
Z2	X1	N+00000001000000
Z2	X2	N+00000002000000
Z2	X3	N+00000012000000
Z3	X1	N+00000002400000
Z3	X2	N+00000002700000
Z3	X4	N+00000001500000
Z3	X5	N+00000000110000
Z3	X6	N+00000005200000

Fichier contenant l'information sur les objectifs:

TEST.FOB

Nom-objet. tr type-optimisation

Z1	N+00000000100000
Z2	N+00000000100000
Z3	N+00000000100000

```
TAPEZ LE NOM DE LA MATRICE DU PROBLEME
 (Maximum 6 caracteres majuscules)     :TEST


TAPEZ LE NOM DU FICHIER QUI CONTIENT (OU QUI CONTIENDRA)
              LES EQUATIONS DU PROBLEME
 (Maximum 6 caracteres majuscules)     :NOURIR


              MANIERE D'INTRODUTRE LES DONNEES

PAR  FICHIER ---> Taper 1

PAR  ECRAN -----> Taper 2

                    Votre choix:1


UTILISATION DU LOGICIEL DE PROGRAMMATION LINEAIRE MULTI-OBJECTIFS
================================================================

POUR RESOUDRE LE PROBLEME ---------------> Taper 1

POUR ACTUALISER LA MATRICE DE DONNEES ---> Taper 2

POUR FINALISER --------------------------> Taper 3

                    Votre choix:1


METHODES POUR RESOUDRE UN PROBLEME DE PROGRAMMATION LINEAIRE MULTIOBJECTIFS
---------------------------------------------------------------------------
1: Critere Global
2: STEM
3: Zionts et Wallenius
4: Geoffrion
5: Goal Programming (Si vous disposez des valeurs cibles pour les objectifs)

          Votre Choix:2


Voulez-vous regarder un exemple de resolution d'un probleme
avec la methode que vous venez de choisir?(Y ou N):N
```

Voulez-vous regarder les solutions Ideales des objectifs?(Y ou N):Y

OBJECTIF	SOL. IDEALE
Z1	+0.22556E+01

ACTIVITE	VALEUR
X1	+0.37829E+01
X2	+0.00000E+00
X3	+0.25000E+00
X4	+0.10000E+02
X5	+0.00000E+00
X6	+0.78618E+00

OBJECTIF	SOL. IDEALE
Z2	+0.17907E+02

ACTIVITE	VALEUR
X1	+0.17907E+01
X2	+0.00000E+00
X3	+0.00000E+00
X4	+0.10000E+02
X5	+0.10000E+02
X6	+0.40000E+01

OBJECTIF	SOL. IDEALE
Z3	+0.15005E+03

ACTIVITE	VALEUR
X1	+0.46686E+01
X2	+0.10000E+01
X3	+0.25000E+00
X4	+0.00000E+00
X5	+0.10000E+02
X6	+0.00000E+00

SOLUTION EFFICACE
==========

OBJECTIF	VALEUR
Z1	+0.3740456E+01
Z2	+0.6963490E+02
Z3	+0.1853898E+03

ACTIVITE	VALEUR
X1	+0.5243670E+01
X2	+0.8599100E+00
X3	+0.0000000E+00
X4	+0.1688280E+01
X5	+0.1000000E+02
X6	+0.0000000E+00

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N

OBJECTIF	SOLUT. IDEALE	SOLUT. ACTUELE
1: (!)Z1	2.25556	3.74046
2: (!)Z2	17.90698	69.63490
3: (!)Z3	150.04651	185.38985

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Voulez-vous relacher(diminuer) un objectif? (Y ou N):Y
Quel objectif?(Tapez le numero correspondant):3
Quelle quantite?:60

SOLUTION EFFICACE
==========

OBJECTIF	VALEUR
Z1	+0.3311551E+01
Z2	+0.5456030E+02
Z3	+0.2453898E+03

ACTIVITE	VALEUR
X1	+0.4474270E+01
X2	+0.4908800E+00
X3	+0.0000000E+00
X4	+0.6245440E+01
X5	+0.1000000E+02
X6	+0.3860000E+00

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N

	OBJECTIF	SOLUT. IDEALE	SOLUT. ACTUELLE	
	========	=============	=============	
1:	(!)Z1	2.25556	3.31155	
2:	(!)Z2	17.90698	54.56030	
3:	(!)Z3	150.04651	245.38984	*

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Voulez-vous relacher(diminuer) un objectif? (Y ou N):Y
Quel objectif?(Tapez le numero correspondant):?
Quelle quantite?:20

SOLUTION EFFICACE
==========

OBJECTIF	VALEUR
Z1	+0.2528518E+01
Z2	+0.6963500E+02
Z3	+0.2453900E+03

ACTIVITE	VALEUR
X1	+0.4003960E+01
X2	+0.4430000E-02
X3	+0.2458900E+00
X4	+0.9211690E+01
X5	+0.1000000E+02
X6	+0.0000000E+00

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N

```
UTILISATION DU LOGICIEL DE PROGRAMMATION LINEAIRE MULTI-OBJECTIFS
=================================================================

POUR RESOUDRE LE PROBLEME ----------------> Taper 1

POUR ACTUALISER LA MATRICE DE DONNEES ---> Taper 2

POUR FINALISER ---------------------------> Taper 3

                  Votre choix:1


METHODES POUR RESOUDRE UN PROBLEME DE PROGRAMMATION LINEAIRE MULTIOBJECTIFS
-------------------------------------------------------------------------
1: Critere Global
2: STEM
3: Zionts et Wallenius
4: Geoffrion
5: Goal Programming (Si vous disposez des valeurs cibles pour les objectifs)

         Votre Choix:4

Voulez-vous regarder un exemple de resolution d'un probleme
avec la methode que vous venez de choisir?(Y ou N):N

Voulez-vous regarder les solutions Ideales des objectifs?(Y ou N):N


             SOLUTION EFFICACE
             =================

        OBJECTIF            VALEUR
        --------            ------
        Z1                +0.3740456E+01
        Z2                +0.6963490E+02
        Z3                +0.1853898E+03


        ACTIVITE            VALEUR
        --------            ------
        X1                +0.5243670E+01
        X2                +0.8599100E+00
        X3                +0.0000000E+00
        X4                +0.1688280E+01
        X5                +0.1000000E+02
        X6                +0.0000000E+00

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N
```

```
            CALCUL DE TAUX DE SUBSTITUTION
            ==============================
    Objectif de reference: Z1

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========  =      ========  =
(!)Z1                    3.740            3.553
(!)Z2                   69.635           73.117
(!)Z3                  185.390          185.390

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:B

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========  =      ========  =
(!)Z1                    3.740            3.553
(!)Z2                   69.635           76.598
(!)Z3                  185.390          185.390

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:I

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========  =      ========  =
(!)Z1                    3.740            3.553
(!)Z2                   69.635           69.635
(!)Z3                  185.390          194.659

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:B

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========  =      ========  =
(!)Z1                    3.740            3.553
(!)Z2                   69.635           69.635
(!)Z3                  185.390          203.929

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:B
```

```
   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========= =      ========= =
(!)Z1                     3.740            3.553
(!)Z2                    69.635           69.635
(!)Z3                   185.390          222.469

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:B

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========= =      ========= =
(!)Z1                     3.740            3.553
(!)Z2                    69.635           69.635
(!)Z3                   185.390          259.546

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:I


     OBJECTIF        TAUX DE SUBSTITUTION
     --------        --------------------
     Z1                         1.000
     Z2                         0.027
     Z3                         0.003


*************************************************************************
* OBJECTIF *    1     *    2     *    3     *    4     *    5     *    6     *
*-----------------------------------------------------------------------*
*Z1        *    3.74  *    3.50  *    3.26  *    3.02  *    2.78  *    2.54  *
*Z2        *   69.63  *   60.46  *   51.29  *   42.11  *   32.94  *   23.77  *
*Z3        *  185.39  *  230.22  *  275.05  *  319.87  *  364.70  *  409.53  *
*-----------------------------------------------------------------------*

Quelle solution(vecteur colonne) preferez-vous?
tapez le numero de la colonne correspondante: 2

Vous desirez continuer les iterations?(taper Y)
Dans le cas contraire, pour finir, taper N
        Votre choix:Y
```

SOLUTION EFFICACE
==========

OBJECTIF	VALEUR
Z1	+0.3499967E+01
Z2	+0.6046100E+02
Z3	+0.2302180E+03

ACTIVITE	VALEUR
X1	+0.4670244E+01
X2	+0.6879280E+00
X3	+0.0000000E+00
X4	+0.3277208E+01
X5	+0.8000000E+01
X6	+0.8000000E+00

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N

CALCUL DE TAUX DE SUBSTITUTION
==========

Objectif de reference: Z1

OBJECTIF	SOLUTION A	SOLUTION B
(!)Z1	3.500	3.325
(!)Z2	60.461	63.484
(!)Z3	230.218	230.218

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
Votre choix:B

OBJECTIF	SOLUTION A	SOLUTION B
(!)Z1	3.500	3.325
(!)Z2	60.461	66.507
(!)Z3	230.218	230.218

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
Votre choix:B

```
   OBJECTIF            SOLUTION A       SOLUTION B
   ========            ========  =      ========  =
(!)Z1                       3.500            3.325
(!)Z2                      60.461           72.553
(!)Z3                     230.218          230.218

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:I

   OBJECTIF            SOLUTION A       SOLUTION B
   ========            ========  =      ========  =
(!)Z1                       3.500            3.325
(!)Z2                      60.461           60.461
(!)Z3                     230.218          241.729

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:B

   OBJECTIF            SOLUTION A       SOLUTION B
   ========            ========  =      ========  =
(!)Z1                       3.500            3.325
(!)Z2                      60.461           60.461
(!)Z3                     230.218          253.240

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:I

     OBJECTIF        TAUX DE SUBSTITUTION
     --------        --------------------
     Z1                             1.000
     Z2                             0.014
     Z3                             0.008

*****************************************************************************
* OBJECTIF *    1    *    2    *    3    *    4    *    5    *    6    *
*----------*---------*---------*---------*---------*---------*---------*
*Z1        *    3.50 *    3.27 *    3.04 *    2.81 *    2.58 *    2.36 *
*Z2        *   60.46 *   62.94 *   65.42 *   67.91 *   70.39 *   72.87 *
*Z3        *  230.22 *  234.86 *  239.51 *  244.16 *  248.81 *  253.45 *
*----------*---------*---------*---------*---------*---------*---------*

Quelle solution(vecteur colonne) preferez-vous?
tapez le numero de la colonne correspondante: 5

Vous desirez continuer les iterations?(taper Y)
Dans le cas contraire, pour finir, taper N
        Votre choix:N
```

SOLUTION EFFICACE
==========

OBJECTIF	VALEUR
Z1	+0.2584030E+01
Z2	+0.7038836E+02
Z3	+0.2488054E+03

ACTIVITE	VALEUR
X1	+0.4266145E+01
X2	+0.1863456E+00
X3	+0.2000000E+00
X4	+0.8655442E+01
X5	+0.2940856E+01
X6	+0.1600000E+00

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N

```
UTILISATION DU LOGICIEL DE PROGRAMMATION LINEAIRE MULTI-OBJECTIFS
================================================================

POUR RESOUDRE LE PROBLEME ----------------> Taper 1

POUR ACTUALISER LA MATRICE DE DONNEES ---> Taper 2

POUR FINALISER ---------------------------> Taper 3

                  Votre choix:1


METHODES POUR RESOUDRE UN PROBLEME DE PROGRAMMATION LINEAIRE MULTIOBJECTIFS
------------------------------------------------------------------------
1: Critere Global
2: STEM
3: Zionts et Wallenius
4: Geoffrion
5: Goal Programming (Si vous disposez des valeurs cibles pour les objectifs)

        Votre Choix:5

Voulez-vous regarder un exemple de resolution d'un probleme
avec la methode que vous venez de choisir?(Y ou N):N

        Valeurs des cibles pour chaque objectif
        =======================================

Tapez la cible pour l'objectif Z1          :1

Tapez la cible pour l'objectif Z2          :10

Tapez la cible pour l'objectif Z3          :100


             SOLUTION EFFICACE
             =================

        OBJECTIF          VALEUR
        --------          ------
        Z1              +0.3884791E+01
        Z2              +0.7411960E+02
        Z3              +0.1682818E+03


        ACTIVITE          VALEUR
        --------          ------
        X1              +0.5449560E+01
        X2              +0.9812000E+00
        X3              +0.0000000E+00
        X4              +0.0000000E+00
        X5              +0.1000000E+02
        X6              +0.0000000E+00

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N

Pourriez-vous donner les poids pour chaque objectif?(Y ou N):N
```

CALCUL DE TAUX DE SUBSTITUTION
==========

Objectif de reference: Z1

OBJECTIF	SOLUTION A	SOLUTION B
(!)Z1	3.885	3.691
(!)Z2	74.120	77.826
(!)Z3	168.282	168.282

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
Votre choix:A

OBJECTIF	SOLUTION A	SOLUTION B
(!)Z1	3.885	3.691
(!)Z2	74.120	75.973
(!)Z3	168.282	168.282

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
Votre choix:I

OBJECTIF	SOLUTION A	SOLUTION B
(!)Z1	3.885	3.691
(!)Z2	74.120	74.120
(!)Z3	168.282	176.696

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
Votre choix:B

OBJECTIF	SOLUTION A	SOLUTION B
(!)Z1	3.885	3.691
(!)Z2	74.120	74.120
(!)Z3	168.282	185.110

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
Votre choix:B

```
   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========= =      ========= =
(!)Z1                      3.885            3.691
(!)Z2                     74.120           74.120
(!)Z3                    168.282          201.938

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:B

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========= =      ========= =
(!)Z1                      3.885            3.691
(!)Z2                     74.120           74.120
(!)Z3                    168.282          235.595

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:B

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========= =      ========= =
(!)Z1                      3.885            3.691
(!)Z2                     74.120           74.120
(!)Z3                    168.282          302.907

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:A

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========= =      ========= =
(!)Z1                      3.885            3.691
(!)Z2                     74.120           74.120
(!)Z3                    168.282          269.251

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:A

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========= =      ========= =
(!)Z1                      3.885            3.691
(!)Z2                     74.120           74.120
(!)Z3                    168.282          252.423

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:A
```

```
   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========= =      ========= =
(!)Z1                     3.885            3.691
(!)Z2                    74.120           74.120
(!)Z3                   168.282          244.009

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:I


     OBJECTIF         TAUX DE SUBSTITUTION
     --------         --------------------
     Z1                          1.000
     Z2                          0.105
     Z3                          0.003


*************************************************************************
* OBJECTIF *     1     *     2     *     3     *     4     *     5     *     6
*-----------------------------------------------------------------------
*Z1        *    3.88 *    3.70 *    3.51 *    3.32 *    3.13 *    2.94
*Z2        *   74.12 *   62.88 *   51.63 *   40.39 *   29.15 *   17.91
*Z3        *  168.28 *  217.02 *  265.76 *  314.50 *  363.24 *  411.98
*-----------------------------------------------------------------------

Quelle solution(vecteur colonne) preferez-vous?
tapez le numero de la colonne correspondante: 2

Vous desirez continuer les iterations?(taper Y)
Dans le cas contraire, pour finir, taper N
        Votre choix:Y


             SOLUTION EFFICACE
             =================

        OBJECTIF            VALEUR
        --------            ------
        Z1                +0.3696414E+01
        Z2                +0.6287708E+02
        Z3                +0.2170208E+03


        ACTIVITE            VALEUR
        --------            ------
        X1                +0.4717788E+01
        X2                +0.7849600E+00
        X3                +0.0000000E+00
        X4                +0.2000000E+01
        X5                +0.1000000E+02
        X6                +0.8000000E+00

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N
Pourriez-vous donner les poids pour chaque objectif?(Y ou N):N
```

CALCUL DE TAUX DE SUBSTITUTION

Objectif de reference: Z1

OBJECTIF	SOLUTION A	SOLUTION B
(!)Z1	3.696	3.512
(!)Z2	62.877	66.021
(!)Z3	217.021	217.021

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
Votre choix:B

OBJECTIF	SOLUTION A	SOLUTION B
(!)Z1	3.696	3.512
(!)Z2	62.877	60.165
(!)Z3	217.021	217.021

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
Votre choix:I

OBJECTIF	SOLUTION A	SOLUTION B
(!)Z1	3.696	3.512
(!)Z2	62.877	62.877
(!)Z3	217.021	227.872

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
Votre choix:B

OBJECTIF	SOLUTION A	SOLUTION B
(!)Z1	3.696	3.512
(!)Z2	62.877	62.877
(!)Z3	217.021	238.723

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
Votre choix:B

```
   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========= =      ========= =
(!)Z1                     3.696            3.512
(!)Z2                    62.877           62.877
(!)Z3                   217.021          260.425

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:I

     OBJECTIF        TAUX DE SUBSTITUTION
     --------        --------------------
     Z1                         1.000
     Z2                         0.029
     Z3                         0.004


*************************************************************************
* OBJECTIF *    1     *    2     *    3     *    4     *    5     *    6
*-----------------------------------------------------------------------
*Z1        *    3.70 *    3.46 *    3.23 *    3.00 *    2.76 *    2.53
*Z2        *   62.88 *   55.24 *   47.59 *   39.95 *   32.31 *   24.67
*Z3        *  217.02 *  254.52 *  292.01 *  329.51 *  367.00 *  404.50
*-----------------------------------------------------------------------

Quelle solution(vecteur colonne) preferez-vous?
tapez le numero de la colonne correspondante: 2

Vous desirez continuer les iterations?(taper Y)
Dans le cas contraire, pour finir, taper N
        Votre choix:Y


            SOLUTION EFFICACE
            =================

        OBJECTIF          VALEUR
        --------          ------
        Z1              +0.3463436E+01
        Z2              +0.5523588E+02
        Z3              +0.2545158E+03


        ACTIVITE          VALEUR
        --------          ------
        X1              +0.4267652E+01
        X2              +0.6279680E+00
        X3              +0.0000000E+00
        X4              +0.3600000E+01
        X5              +0.8000000E+01
        X6              +0.1391096E+01

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N

Pourriez-vous donner les poids pour chaque objectif?(Y ou N):N
```

```
                CALCUL DE TAUX DE SUBSTITUTION
                ==============================
    Objectif de reference: Z1

   OBJECTIF              SOLUTION A       SOLUTION B
   ========              ========  =      ========  =
(!)Z1                         3.463            3.290
(!)Z2                        55.236           57.999
(!)Z3                       254.516          254.516

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
         Votre choix:B

   OBJECTIF              SOLUTION A       SOLUTION B
   ========              ========  =      ========  =
(!)Z1                         3.463            3.290
(!)Z2                        55.236           60.759
(!)Z3                       254.516          254.516

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
         Votre choix:B

   OBJECTIF              SOLUTION A       SOLUTION B
   ========              ========  =      ========  =
(!)Z1                         3.463            3.290
(!)Z2                        55.236           66.283
(!)Z3                       254.516          254.516

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
         Votre choix:B

   OBJECTIF              SOLUTION A       SOLUTION B
   ========              ========  =      ========  =
(!)Z1                         3.463            3.290
(!)Z2                        55.236           77.330
(!)Z3                       254.516          254.516

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
         Votre choix:I
```

```
   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========  =      ========  =
(!)Z1                     3.463            3.290
(!)Z2                    55.236           55.236
(!)Z3                   254.516          267.242

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:A

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========  =      ========  =
(!)Z1                     3.463            3.290
(!)Z2                    55.236           55.236
(!)Z3                   254.516          260.879

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:A

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========  =      ========  =
(!)Z1                     3.463            3.290
(!)Z2                    55.236           55.236
(!)Z3                   254.516          257.697

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:A

   OBJECTIF          SOLUTION A       SOLUTION B
   ========          ========  =      ========  =
(!)Z1                     3.463            3.290
(!)Z2                    55.236           55.236
(!)Z3                   254.516          256.107

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

Quelle solution(vecteur) preferez-vous?(A ou B)
Si vous etes indifferent, taper I
        Votre choix:I

     OBJECTIF        TAUX DE SUBSTITUTION
     --------        --------------------
     Z1                             1.000
     Z2                             0.008
     Z3                             0.109
```

OBJECTIF	1	2	3	4	5	6
Z1	3.46	3.41	3.35	3.30	3.25	3.19
Z2	55.24	64.28	73.32	82.36	91.40	100.44
Z3	254.52	234.05	213.58	193.12	172.65	152.19

Quelle solution(vecteur colonne) preferez-vous?
tapez le numero de la colonne correspondante: 3

Vous desirez continuer les iterations?(taper Y)
Dans le cas contraire, pour finir, taper N
Votre choix:N

SOLUTION EFFICACE
==========

OBJECTIF	VALEUR
Z1	+0.3354276E+01
Z2	+0.7331753E+02
Z3	+0.2135848E+03

ACTIVITE	VALEUR
X1	+0.4735615E+01
X2	+0.6980688E+00
X3	+0.1000000E+00
X4	+0.2160000E+01
X5	+0.4800000E+01
X6	+0.8346576E+00

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N

UTILISATION DU LOGICIEL DE PROGRAMMATION LINEAIRE MULTI-OBJECTIFS
==========

POUR RESOUDRE LE PROBLEME ----------> Taper 1

POUR ACTUALISER LA MATRICE DE DONNEES ---> Taper 2

POUR FINALISER ----------> Taper 3

Votre choix:1

METHODES POUR RESOUDRE UN PROBLEME DE PROGRAMMATION LINEAIRE MULTIOBJECTIFS

1: Critere Global
2: STEM
3: Zionts et Wallenius
4: Geoffrion
5: Goal Programming (Si vous disposez des valeurs cibles pour les objectifs)

Votre Choix:3

Voulez-vous regarder un exemple de resolution d'un probleme
avec la methode que vous venez de choisir?(Y ou N):N

SOLUTION EFFICACE
==========

OBJECTIF	VALEUR
Z1	+0.3884791E+01
Z2	+0.7411960E+02
Z3	+0.1682818E+03

ACTIVITE	VALEUR
X1	+0.5449560E+01
X2	+0.9812000E+00
X3	+0.0000000E+00
X4	+0.0000000E+00
X5	+0.1000000E+02
X6	+0.0000000E+00

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N

Voulez-vous continuer les iterations?(Y ou N):Y

Pour le Taux de substitution suivant:

OBJECTIF	SOLUTION ACTUELLE	TAUX DE SUBSTITUTION
(!)Z1	3.88479	+2.31302
(!)Z2	74.11960	-81.27680
(!)Z3	168.28184	+66.03468

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

1: Vous l'acceptez
2: Vous ne l'acceptez pas
3: Vous etes indifferent

Votre choix:2

Pour le suivant Taux de substitution

OBJECTIF	SOLUTION ACTUELLE	TAUX DE SUBSTITUTION
(!)Z1	3.88479	+0.08549
(!)Z2	74.11960	+2.65638
(!)Z3	168.28184	-10.13338

(!)---> L'objectif dont le nom est precede par ce signal est a MINIMISER

1: Vous l'acceptez
2: Vous ne l'acceptez pas
3: Vous etes indifferent

Votre choix:1

SOLUTION EFFICACE

OBJECTIF	VALEUR
Z1	+0.2942907E+01
Z2	+0.1790700E+02
Z3	+0.4119768E+03

ACTIVITE	VALEUR
X1	+0.1790700E+01
X2	+0.0000000E+00
X3	+0.0000000E+00
X4	+0.1000000E+02
X5	+0.1000000E+02
X6	+0.4000000E+01

Voulez-Vous avoir ces resultats sur l'imprimante?(Y ou N):N

Voulez-vous continuer les iterations?(Y ou N):N

CONCLUSIONS

Cette partie du mémoire a pour but essentiel de dresser un bilan général sur le travail réalisé.

Nous parlerons d'abord des difficultés rencontrées au cours de la réalisation et ensuite des possibilités d'amélioration et d'extension.

Même si le but proposé, qui était de réaliser un logiciel interactif de programmation linéaire multiobjectifs, a été atteint, il faut cependant souligner les limites du logiciel et les difficultés rencontrées au long du développement de ce travail.

Au départ, nous ne disposions pas d'une version en mode batch du logiciel Lamps, permettant d'utiliser ses modules comme des sous-routines. Nous avons du investir beaucoup de temps pour connaître les possibilités réelles d'utilisation de ce logiciel, en attendant sa mise au point. C'est uniquement vers la fin du mois de Janvier / 84 que nous avons pu commencer l'implémentation du logiciel de PLMO après la mise au point de certaines routines de Lamps par le Centre de Calcul de l'Institut d'Informatique.

Une fois l'implémentation démarrée, nous avons rencontré une autre difficulté technique: Le système DEC-20 ne permet pas de faire des appels de routines Cobol à partir d'un programme écrit en Fortran; ceci nous a obligé de faire les échanges de données par l'intermédiaire des fichiers, ce qui nuit à la performance de temps d'exécution.

Un autre facteur qui influence le temps d'exécution réside en la non-disponibilité de toutes les routines de logiciel Lamps.

En ce qui concerne les possibilités d'extension et d'amélioration, nous proposons d'implémenter un logiciel de PLS plus approprié aux problèmes de PLMO; c'est-à-dire: disposer de la matrice inverse de la base, conserver les bases antérieures, ce qui permettra de réaliser des analyses de sensibilité et de post-optimisation complètes.

Vu la structure modulaire implementée, il serait facile d'ajouter d'autres méthodes de PLMO, qui n'ont pas été implémentées, d'une part par manque de temps, et d'autre part parce que le logiciel Lamps ne convenait pas à l'implémentation.

Du point de vue dimension des problèmes que l'on peut traiter, nous sommes actuellement limités à 100 variables de décision (sans compter les variables d'écart) et 100 équations (9 fonctions objectifs et 91 contraintes). Mais ceci n'est pas vraiment une contrainte, puisqu'il suffit d'augmenter les dimensions des tableaux définis.

De plus il faut remarquer que les problèmes à traiter doivent garder une dimension telle que l'utilisateur puisse maîtriser le problème.

Pour terminer, nous signalons que les problèmes rencontrés dans la litérature d'Aide à la Décision Multiobjectifs, sont souvent de niveau académique et théorique ce qui relève certainement de la nouveauté du sujet...

BIBLIOGRAPHIE

[1] BENAYOUN R., de MONTGOLFIER J., TERGNY J., LARITCHEV O.
Linear Programming with Multiple Objective Functions: Step Method (STEM).
Mathematical Programming, Vol. 1, No. 3, 1971.

[2] CRAMA Y.
Les solutions du type STEM en Programmation à Objectifs Multiples.
Université de Liège.

[3] de MONTGOLFIER J., BERTIER P.
Approche Multicritère des problèmes de Décision.
Collection afcet, 1978.

[4] de SAMBLANCKX S., DEPRAETERE P., MULLER H.
Critical considerations concerning the multicriteria analysis by the method of Zionts and Wallenius.
European Journal of Operational Research, Vol. 10, No. 1, May 1982.

[5] DYER J. S.
A Time-Sharing Computer Program for the Solution of the Multiple Criteria Problem.
Management Science, Vol. 19, No. 12, August 1973.

[6] DYER J. S.
Interactive Goal Programming.
Management Science, Vol. 19, No. 1, September 1972.

[7] FICHEFET J.
Aide à la Décision Multicritère.
Notes de cours.
Institut d'Informatique, Namur.

[8] FICHEFET J.
Recherche Opérationnelle de Gestion.
Notes de cours.
Institut d'Informatique, Namur.

[9] FICHEFET J.
GPSTEM: An Interactive Multiobjective Optimization Method.
Progress in Operations Research, Vol. 1, North Holland, Amsterdam, 1976.

[10] FICHEFET J.
Implémentation d'un Langage Interactif de Programmation Linéaire Multicritère.
Institut d'Informatique, Namur.

[11] GEOFFRION A. M., DYER J.S., FEINBERG A.
An Interactive Approach for Multi-Criterion Optimization, with an Application to the Operation of an Academic Departement.
Management Science, Vol. 19, No. 4, December 1972.

BIBLIOGRAPHIE

[12] GOICOECHEA A., HANSEN D. R., DUCKSTEIN L.
Multiobjective Decision Analysis with Engineering and Business Applications.
John Wiley and Sons, 1982.

[13] HANSEN P.
Essays and Surveys on Multiple Criteria Decision Making.
Springer-Verlag, 1982.

[14] HWANG C. L., MAZUD A. S. A.
Multiple Objective Decision Making. Methods and applications.
Springer-Verlag, 1979.

[15] IJIRI Y.
Management Goals and Accounting for Control.
North-Holland Publishing Company Amsterdam, 1965.

[16] JOHNSON L.A., MONTGOMERY D.C.
Operations Research in Production Planning, Scheduling, and Invntory Control.
John Wiley & Sons, 1974.

[17] KEENEY R. L., RAIFFA H., MEYER R.F.
Decisions with Multiple Objectives: Preferences and Value Tradeoffs.
John Wiley & Sons, 1976.

[18] LECLERCQ J. P.
Notes de cours (3ème Cycle FNRS, Modelisation des Préférences et Aide à la Décision Multicritère)
Bruxelles, 1984.

[19] ROBERTS F. S.
Measurement Theory.
Addison-Wesley, Reading, Mass, 1979.

[20] THIRIEZ H. ZIONTS S.
Multiple Criteria Decision Making.
Springer Verlag, 1976.

[21] VANGELDERE J.
Quelques remarques en relation avec STEM.
Réunion groupe EURO "Aide à la Décision Multicritère", Bochum, Octobre 1980.
Université de Liège.

[22] VINCKE Ph.
Présentation et analyse de neuf méthodes multicritères interactives.
Cahier No. 42 du LAMSADE, Décembre 1982.

BIBLIOGRAPHIE

[23] VINCKE Ph.
Une Méthode Interactive en Programmation Linéaire a plusieurs Fonctions Economiques.
Revue Francaise d'Automatique, Informatique et Recherche Opérationnelle, Juin 1976.

[24] WALLENIUS J.
Comparative Evaluation of some Interactive Approches to Multicriterion Optimization.
Management Science, Vol. 21, No. 12, August 1975.

[25] ZELENY M.
Linear Multiobjective Programming.
Springer-Verlag, 1974.

[26] ZELENY M.
Multiple Criteria Decision Making.
Springer-Verlag, 1976.

[27] ZIONTS S., WALLENIUS J.
An Interactive Programming Method for Solving the Multiple Criteria Problem.
Management Science, Vol. 22, No. 6, February 1976.

[28] CAP Scientific Ltd.
LAMPS USER GUIDE.
February 1983.

[29] DIGITAL EQUIPEMENT CORPORATION
COBOL-74 Langage Manual.
Juanary 1979.

[30] DIGITAL EQUIPEMENT CORPORATION
FORTRAN Langage Manual.
February 1983.

FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX - NAMUR

INSTITUT D'INFORMATIQUE

UN LOGICIEL INTERACTIF DE PROGRAMMATION MULTIOBJECTIFS

(ANNEXE)

Mémoire présenté par

Victor AGUILAR V.

en vue de l'obtention du grade de

Licencié et Maître en Informatique

Promoteur: J. FICHEFET

ANNEE ACADEMIQUE 1983-1984

P R O G R A M M E S

1.- INTRODUCTION DES DONNEES

2.- TRAITEMENTS

3.- INTERFACE AVEC LAMPS

INTRODUCTION DES DONNEES

```
*=======================*
IDENTIFICATION DIVISION.
*=======================*

PROGRAM-ID. DONNES.
AUTHOR. AGUILAR VICTOR.

*=============================================================================*
* Ce programme sert a l'introduction de donnees correspondant a un    *
* modele mathematique d'un probleme de programmation lineaire multi= *
* -objectifs.                                                         *
*            Les donnees sont introduites sous la "forme courante"    *
* d'une equation(on ecrit uniquement les termes correspondants a des  *
* coefficients non nuls), transformees et stockees dans de fichiers   *
* sous une forme directement utilisable par "Lamps".                  *
*                                                                     *
* La saisie de donnees peut se faire de 2 facons:                     *
* 1: A partir d'un fichier cree prealablement a l'aide d'une editeur  *
*    de texte.                                                        *
* 2: De facon interactive. Dans ce cas, le programme cree en meme     *
*    temps que les fichiers de sortie utilisables par Lamps, un autre *
*    fichier qui contiendra les equations introduites.                *
*    Pour une application ulterieure on pourrait utiliser ce fichier  *
*    directement ou apres modifications par l'intermediaire d'un edi- *
*    teur de texte.                                                   *
*                                                                     *
* Normes concernant l'ecriture des equations a l'ecran:               *
* ----------------------------------------------------                *
* 1) Une equation est une chaine de caracteres terminee par un point  *
*    virgule(";").                                                    *
*    Cette chaine de caracteres se compose de:                        *
*    a) Un nom d'equation.                                            *
*    b) Une suite alternee de termes(coefficient et nom de variable)  *
*       et d'operateurs arithmetique "+" ou "-".                      *
*    c) - Un signe d'egalite ou d'inegalite suivi du terme independent*
*         pour les contraintes.                                       *
*       - Le signe "$" suivi d'un indicateur du type d'optimisation-  *
*         ("max" ou "min") pour les fonctions objectifs.              *
*                                                                     *
* 2) - Les signes ">=" et "<=" sont representes par ">" et "<" respec-*
*      tivement.                                                      *
*    - Les noms d'equations, noms de variables, coefficients, signes- *
*      (+,-,<,>,=,$,min,max) doivent etre separes par au moins un ca- *
*      ractere blanc.                                                 *
*    - Tous les coefficients doivent contenir explicitement le point  *
*      decimal, meme les entiers.                                     *
*    - Les coefficients unitaires doivent etre explicites(1.).        *
*    - La longueur maximale d'un nom de variable ou nom d'equation est*
*      de 12 caracteres.                                              *
*                                                                     *
*=====================================================================*
```

```
*=======================*
ENVIRONMENT DIVISION.
*=======================*

*---------------------*
INPUT-OUTPUT SECTION.
*---------------------*

FILE-CONTROL.
*------------*

      SELECT FICHIER-EXEMPLE ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-LIGNES ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-FONCTION ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-VAR ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-DATA ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-OBJECTIFS ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-TICKET ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-VAR-TRIE ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-OBJECTIFS-TRIE ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-LIGNES-TRIE ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-FON-TRIE ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-TICKET-TRIE ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-TRI-VAR ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-TRI-OBJECTIFS ASSIGN TO DSK
             RECORDING MODE IS ASCII.
      SELECT FICHIER-TRI-LIGNES ASSIGN TO DSK
             RECORDING MODE IS ASCII.
```

```
*===============*
DATA DIVISION.
*===============*

*-------------*
FILE SECTION.
*-------------*

* Fichier contenant un exemple de la facon d'entrer les donnees
FD   FICHIER-EXEMPLE LABEL RECORD IS STANDARD
                     VALUE ID IS "EXEM  PLE".
01   ARTICLE-EXEMPLE                          PIC X(80)        DISPLAY-7.

* Fichier contenant les donnees brutes(sous forme d'equations)
FD   FICHIER-DATA LABEL RECORD IS STANDARD
                  VALUE ID IS NOM-FICHIER.
01   ARTICLE-DATA                             PIC X(80)        DISPLAY-7.

* Fichier contenant le nom de la matrice de donnees et une
* indication auprogramme principal precisant si la saisie
* des donnees s'est bien realisee
FD   FICHIER-TICKET LABEL RECORD IS STANDARD
                    VALUE ID IS FICHIER-D.
01   ARTICLE-TICKET.
     02 NOM-FICHIER-DATA                      PIC X(8)         DISPLAY-7.
     02 FILLER                                PIC X(4)         DISPLAY-7.
     02 CHOIX                                 PIC X            DISPLAY-7.
     02 ETAT-FICHIER                          PIC X(2)         DISPLAY-7.
     02 FILLER                                PIC X(13)        DISPLAY-7.

* Fichier contenant les coefficients de la matrice de donnees
FD   FICHIER-VAR LABEL RECORD IS STANDARD
                 VALUE ID IS FICHIER-V.
01   ARTICLE-VAR.
     02 NUM-LIGNE-VAR                         PIC X(12)        DISPLAY-7.
     02 NUM-VARIABLE                          PIC X(12)        DISPLAY-7.
     02 RELATION-VAR                          PIC X            DISPLAY-7.
     02 VALEUR-COEFFICIENT                    PIC S9(9)V9(5)   DISPLAY-7
        SIGN IS LEADING SEPARATE.

* Fichier contenant les noms de fonctions objectifs
* et le type d'optimisation
FD   FICHIER-FONCTOBJ LABEL RECORD IS STANDARD
                      VALUE ID IS FICHIER-F.
01   ARTICLE-FONCTOBJ                         PIC X(28)        DISPLAY-7.

* Fichier contenant les noms de toutes les equations de la
* matrice de connees, et la valeur du terme independant
FD   FICHIER-LIGNES LABEL RECORD IS STANDARD
                    VALUE ID IS FICHIER-L.
01   ARTICLE-LIGNES.
     02 NUM-LIGNE                             PIC X(12)        DISPLAY-7.
     02 RELATION                              PIC X            DISPLAY-7.
     02 VALEUR-RESSOURCE                      PIC S9(9)V9(5)   DISPLAY-7
        SIGN IS LEADING SEPARATE.
```

```
* Fichier contenant les coefficients des variables
* des fonctions objectifs
FD   FICHIER-OBJECTIFS LABEL RECORD IS STANDARD
                       VALUE ID IS FICHIER-O.
01   ARTICLE-OBJECTIF                    PIC X(40)        DISPLAY-7.

FD   FICHIER-VAR-TRIE LABEL RECORD IS STANDARD
                      VALUE ID IS FICHIER-V.
01   ARTICLE-VAR-TRIE                    PIC X(40)        DISPLAY-7.

FD   FICHIER-LIGNES-TRIE LABEL RECORD IS STANDARD
                         VALUE ID IS FICHIER-L.
01   ARTICLE-LIGNES-TRIE                 PIC X(28)        DISPLAY-7.

FD   FICHIER-OBJECTIFS-TRIE LABEL RECORD IS STANDARD
                            VALUE ID IS FICHIER-O.
01   ARTICLE-OBJECTIF-TRIE               PIC X(40)        DISPLAY-7.

FD   FICHIER-TICKET-TRIE LABEL RECORD IS STANDARD
                         VALUE ID IS FICHIER-D.
01   ARTICLE-TICKET-TRIE PIC X(29)                        DISPLAY-7.

FD   FICHIER-FOB-TRIE LABEL RECORD IS STANDARD
                      VALUE ID IS FICHIER-F.
01   ARTICLE-FOB-TRIE PIC X(28)                           DISPLAY-7.

SD   FICHIER-TRI-VAR.
01   ARTICLE-TRI-VAR.
     02 NUM-TRI-LIG                       PIC X(12)        DISPLAY-7.
     02 NUM-TRI                           PIC X(12)        DISPLAY-7.
     02 FILLER                            PIC X(16)        DISPLAY-7.

SD   FICHIER-TRI-OBJECTIFS.
01   ARTICLE-TRI-OBJECTIF.
     02 LIGNE-O-TRI                       PIC X(12)        DISPLAY-7.
     02 NOM-VAR-O-TRI                     PIC X(12)        DISPLAY-7.
     02 FILLER                            PIC X(16)        DISPLAY-7.

SD   FICHIER-TRI-LIGNES.
01   ARTICLE-TRI-LIGNES.
     02 LIGNE-L-TRI                       PIC X(12)        DISPLAY-7.
     02 RELATION-TRI                      PIC X            DISPLAY-7.
     02 FILLER                            PIC X(15)        DISPLAY-7.
```

```
*-----------------------*
WORKING-STORAGE SECTION.
*-----------------------*

77    FIN-FICHIER                          PIC X(3)          DISPLAY-7.
77    FIN-ARTICLE                          PIC X(3)          DISPLAY-7.
77    BON-NOM                              PIC X(3)          DISPLAY-7.
77    BONNE-EQUATION                       PIC X(3)          DISPLAY-7.
77    OPTIMISATION                         PIC X(3)          DISPLAY-7.
77    SYMBOLE                              PIC X             DISPLAY-7.
77    SIGNE-IT                             PIC X             DISPLAY-7.
77    REPONSE                              PIC Y             DISPLAY-7.
77    SW                                   PIC 9             COMP.
77    I                                    PIC 9(3)          COMP.
77    LN                                   PIC 9(4)          COMP.
77    L1                                   PIC 9(4)          COMP.
77    ENCORE                               PIC 9             COMP.
77    INDICATEUR                           PIC 9             COMP.
77    COMPTEUR-LIGNES                      PIC 9(3)          COMP.
77    COMPTEUR-VAR                         PIC 9(3)          COMP.
77    COMPTEUR-EQUATIONS                   PIC 9(3)          COMP.
77    POINTEUR-CHAINE                      PIC 9(4)          COMP.
77    POINTEUR-QUANTITE                    PIC 9(2)          COMP.
77    DECI                                 PIC 9(5)          COMP.
77    NOM                                  PIC X(12)         DISPLAY-7.
77    RESOURCE                             PIC X(15)         DISPLAY-7.

****** Items dont on peut changer la valeur si besoin en est ******
77    MAXIMUM-LIGNES                       PIC 9(3)          VALUE 25.
77    MAXIMUM-VAR                          PIC 9(3)          VALUE 100.
77    MAXIMUM-EQUATIONS                    PIC 9(3)          VALUE 100.
01    RECORD-TEST.
      02 ITEM OCCURS 25 TIMES              PIC X(80)         DISPLAY-7.
01    RECORD-AUX REDEFINES RECORD-TEST.
      02 LIGNE-STRING OCCURS 2000 TIMES    PIC X             DISPLAY-7.
01    TABLE-NOMS                           PIC X(12)         DISPLAY-7
      OCCURS 1 TO 100 TIMES
      DEPENDING ON COMPTEUR-EQUATIONS
      INDEXED BY K1.
01    TABLE-AUX OCCURS 100 TIMES.
      02 NUMERO-T                          PIC X(15)         DISPLAY-7.
      02 SIGNE-T                           PIC X             DISPLAY-7.
      02 NOM-T                             PIC X(12)         DISPLAY-7.
*************************************************************************
```

```
01   LIGNE-ECRAN                          PIC X(80)          DISPLAY-7.
01   LIGNE-CHAR REDEFINES LIGNE-ECRAN.
     02 CARACTER                          PIC X              DISPLAY-7
        OCCURS 80 TIMES INDEXED BY K2.
01   QUANTITE.
     02 ENTIER                            PIC 9(9).
     02 DECI-1                            PIC V9(5).
01   Q-AUX REDEFINES QUANTITE             PIC S9(9)V9(5).
01   NUMERO.
     02 NUMERO-TEST                       PIC X              DISPLAY-7
        OCCURS 1 TO 15 DEPENDING ON LN
                       INDEXED BY K3.
01   NOM-FICHIER.
     02 NOM-F                             PIC X(6)           DISPLAY-7.
     02 EXTENTION                         PIC X(3)           DISPLAY-7
                                                            VALUE "DAT".

01   FICHIER-D.
     02 NOM-F-D                           PIC X(6)           DISPLAY-7.
     02 EXT-D                             PIC X(3)           DISPLAY-7
                                                            VALUE "MAT".

01   FICHIER-L.
     02 NOM-F-L                           PIC X(6)           DISPLAY-7.
     02 EXT-L                             PIC X(3)           DISPLAY-7
                                                            VALUE "LIC".

01   FICHIER-V.
     02 NOM-F-V                           PIC X(6)           DISPLAY-7.
     02 EXT-V                             PIC X(3)           DISPLAY-7
                                                            VALUE "VAR".

01   FICHIER-O.
     02 NOM-F-O                           PIC X(6)           DISPLAY-7.
     02 EXT-O                             PIC X(3)           DISPLAY-7
                                                            VALUE "OBJ".

01   FICHIER-F.
     02 NOM-F-F                           PIC X(6)           DISPLAY-7.
     02 EXT-F                             PIC X(3)           DISPLAY-7
                                                            VALUE "FOB".

*---------------*
LINKAGE SECTION.
*---------------*

77   NOM-FICHIERS                        PIC X(6)           DISPLAY-7.
77   NOM-FICHIER-EO                      PIC X(6)           DISPLAY-7.
```

```
*=================================================*
PROCEDURE DIVISION USING NOM-FICHIER-EQ NOM-FICHIERS.
*=================================================*

*+++++++++++++++++*
DECISION SECTION.
*+++++++++++++++++*

     MOVE NOM-FICHIERS TO NOM-F-D, NOM-F-L, NOM-F-V, NOM-F-O, NOM-F-F.
     MOVE NOM-FICHIER-EQ TO NOM-F.
     OPEN I-O FICHIER-TICKET.
     READ FICHIER-TICKET
          AT END DISPLAY "Pas nom de fichier"
                                  MOVE 1 TO SW.
     IF CHOIX = 1
        PERFORM LECTURE-DES-FICHIER
     ELSE IF CHOIX = 2
        PERFORM LECTURE-DES-ECRAN.
     EXIT PROGRAM.

*++++ PROCEDURE DE LECTURE D'APRES UN FICHIER++++*
LECTURE-DES-FICHIER SECTION.
*++++++++++++++++++++++++++++++++++++++++++++++++*

*
     PERFORM INITIALISATIONS.
*
     PERFORM AGROUPEMENT-ECLATEMENT UNTIL FIN-FICHIER = "OUI"
                                                 OR SW = 1.
*
     PERFORM CLOTURES.
```

```
*-----------------*
NIVEAU-11 SECTION.
*-----------------*

  INITIALISATIONS.
* --------------- *
     MOVE "NON" TO FIN-FICHIER.
     MOVE 0 TO SW.
     MOVE 1 TO COMPTEUR-EQUATIONS.
     OPEN INPUT FICHIER-DATA
          OUTPUT FICHIER-LIGNES,FICHIER-FONCTOBJ
                 FICHIER-VAR,FICHIER-OBJECTIFS.
     PERFORM LECTURE-FICH.

  AGROUPEMENT-ECLATEMENT.
* --------------------- *
     MOVE 0 TO COMPTEUR-LIGNES.
     MOVE "NON" TO FIN-ARTICLE.
     PERFORM GENERATION-ARTICLE-GROUPE UNTIL FIN-ARTICLE = "OUI"
                                             OR SW = 1.
     PERFORM PDECOMPOSITION-ARTICLE-GROUPE.
     PERFORM PTEST-NOMBRE-EQUATIONS.
     PERFORM PECRITURE-FICHIER-LIGNES.
     PERFORM LECTURE-FICH.
     ADD 1 TO COMPTEUR-EQUATIONS.

  CLOTURES.
* ------- *
     PERFORM PERREUR-FIN.
     REWRITE ARTICLE-TICKET.
     CLOSE FICHIER-TICKET FICHIER-DATA FICHIER-FONCTOBJ
           FICHIER-LIGNES FICHIER-VAR FICHIER-OBJECTIFS.
     SORT FICHIER-TRI-LIGNES DESCENDING KEY RELATION-TRI
          USING FICHIER-TICKET
          GIVING FICHIER-TICKET-TRIE.
     PERFORM PSORTING-FICHIER-VAR-OBJ.
     PERFORM PSORTING-FICHIER-LIGNES.
```

```
*----------------*
NIVEAU-12 SECTION.
*----------------*

  LECTURE-FICH.
* ---------- *
     IF CHOIX = 1
        READ FICHIER-DATA INTO LIGNE-ECRAN
             AT END MOVE "OUI" TO FIN-FICHIER
     ELSE ACCEPT LIGNE-ECRAN.

  GENERATION-ARTICLE-GROUPE.
* ------------------------ *
     ADD 1 TO COMPTEUR-LIGNES.
     PERFORM TEST-LIGNES.
     PERFORM PPASSAGE.

  PDECOMPOSITION-ARTICLE-GROUPE.
* ---------------------------- *
     IF SW = 0
        PERFORM DECOMPOSITION.

  PTEST-NOMBRE-EQUATIONS.
* --------------------- *
     IF SW = 0
        PERFORM TEST-NOMBRE-EQUATIONS.

  PECRITURE-FICHIER-LIGNES.
* ----------------------- *
     IF SW = 0
        PERFORM ECRITURE-LIGNES.

  PERREUR-FIN.
* ---------- *
     IF SW = 0
        MOVE "OK" TO ETAT-FICHIER
     ELSE MOVE "NO" TO ETAT-FICHIER.

  PSORTING-FICHIER-VAR-OBJ.
* ----------------------- *
     IF SW = 0
        PERFORM SORTING-FICHIER-VAR-OBJ.

  PSORTING-FICHIER-LIGNES.
* ---------------------- *
     IF SW = 0
        PERFORM SORTING-FICHIER-LIGNES.
```

```
*----------------*
NIVEAU-13 SECTION.
*----------------*

  TEST-LIGNES.
* ---------- *
     IF COMPTEUR-LIGNES > MAXIMUN-LIGNES
        MOVE 1 TO SW
        DISPLAY "On depasse le nombre de lignes pour une equation".

  PPASSAGE.
* ------- *
     IF SW = 0
        PERFORM PASSAGE-TEST.

  DECOMPOSITION.
* ------------ *
     MOVE 1 TO POINTEUR-CHAINE.
     UNSTRING RECORD-TEST DELIMITED BY ALL " "
              INTO NOM
              WITH POINTER POINTEUR-CHAINE.
     PERFORM TEST-NOM-LIGNE.
     PERFORM PCORPS-DECOMPOSITION.
     PERFORM PSYMBOLE-TI.
     PERFORM PECRITURE-FICHIER-VAR.

  TEST-NOMBRE-EQUATIONS.
* -------------------- *
     IF COMPTEUR-EQUATIONS > MAXIMUN-EQUATIONS
        MOVE 1 TO SW
        DISPLAY "On depasse le nombre maximun d'equations permi"

     ELSE MOVE NOM TO TABLE-NOMS (COMPTEUR-EQUATIONS).

  ECRITURE-LIGNES.
* -------------- *
     MOVE NOM TO NOM-LIGNE.
     IF SYMBOLE = "=" MOVE "N" TO RELATION
     ELSE IF SYMBOLE = ">" MOVE "G" TO RELATION
          ELSE IF SYMBOLE = "<" MOVE "L" TO RELATION
               ELSE MOVE "E" TO RELATION.
     IF RELATION = "N" WRITE ARTICLE-FONCTOBJ FROM ARTICLE-LIGNES
     ELSE WRITE ARTICLE-LIGNES.

  SORTING-FICHIER-VAR-OBJ.
* ---------------------- *
     DISPLAY "Je travaille, attendez un moment"
     SORT FICHIER-TRI-VAR ASCENDING KEY NOM-TRI NOM-TRI-LIG
          USING FICHIER-VAR
          GIVING FICHIER-VAR-TRIE.
     SORT FICHIER-TRI-OBJECTIFS ASCENDING KEY
                                LIGNE-O-TRI NOM-VAR-O-TRI
          USING FICHIER-OBJECTIFS
          GIVING FICHIER-OBJECTIFS-TRIE.
```

```
  SORTING-FICHIER-LIGNES.
* --------------------- *
     SORT FICHIER-TRI-LIGNES ASCENDING KEY LIGNE-L-TRI
          USING FICHIER-LIGNES
          GIVING FICHIER-LIGNES-TRIE.
     SORT FICHIER-TRI-LIGNES ASCENDING KEY LIGNE-L-TRI
          USING FICHIER-FONCTOBJ
          GIVING FICHIER-FOB-TRIE.

*-----------------*
NIVEAU-14 SECTION.
*-----------------*

  PASSAGE-TEST.
* ----------- *
     MOVE LIGNE-ECRAN TO ITEM (COMPTEUR-LIGNES).
     PERFORM TEST-FIN-ARTICLE-GROUPE.

  TEST-NOM-LIGNE.
* ------------- *
     SET K1 TO 1.
     SEARCH TABLE-NOMS VARYING K1 AT END MOVE 0 TO SW
            WHEN TABLE-NOMS (K1) = NOM MOVE 1 TO SW
                 DISPLAY "Nom de l'equation dupliquee:", NOM.

  PCORPS-DECOMPOSITION.
* ------------------- *
     IF SW = 0
        PERFORM CORPS-DECOMPOSITION.

  PSYMBOLE-TI.
* ---------- *
     IF SW = 0
        PERFORM TRAITEMENT-SYMBOLE-ET-TI.

  PECRITURE-FICHIER-VAR.
* -------------------- *
     IF SW = 0
        PERFORM ECRITURE-FICHIER-VAR.
```

```
*----------------*
NIVEAU-15 SECTION.
*----------------*

  TEST-FIN-ARTICLE-GROUPE.
* ---------------------- *
     SET K2 TO 1.
     SEARCH CARACTER VARYING K2 AT END PERFORM LECTURE-FICH
            WHEN CARACTER (K2) = ";" MOVE "OUI" TO FIN-ARTICLE.

  CORPS-DECOMPOSITION.
* ------------------- *
     MOVE 0 TO COMPTEUR-VAR.
     PERFORM TRAITEMENT-ARTICLE-GROUPE UNTIL
             (LIGNE-STRING(POINTEUR-CHAINE) = "S"
              OR = ">" OR = "<" OR = "=" OR = ";") OR SW = 1.

  TRAITEMENT-SYMBOLE-ET-TI.
* ------------------------ *
     MOVE LIGNE-STRING (POINTEUR-CHAINE) TO SYMBOLE.
     PERFORM PLUS-UNE.
     PERFORM PLUS-UNE UNTIL LIGNE-STRING (POINTEUR-CHAINE) NOT = " ".
     IF SYMBOLE = ";"
        DISPLAY "Il faut un blanc entre nom et signe dans l'equation: "
                NOM
        MOVE 1 TO SW
     ELSE IF SYMBOLE = "S"
             PERFORM TYPE-OPTIMISATION
             PERFORM PTEST-PVERGULE
          ELSE PERFORM TERME-INDEPENDENTE.
     IF SW = 0
        MOVE Q-AUX TO VALEUR-RESOURCE.

  ECRITURE-FICHIER-VAR.
* -------------------- *
     PERFORM MOVE-A-ARTICLE-VAR VARYING I FROM 1 BY 1
                                UNTIL I > COMPTEUR-VAR.
```

```
*-----------------*
NIVEAU-16 SECTION.
*-----------------*

  TRAITEMENT-ARTICLE-GROUPE.
* ------------------------ *
     ADD 1 TO COMPTEUR-VAR.
     PERFORM TEST-VARIABLE.
     PERFORM PMOUVEMENTS.
     PERFORM PTEST-SIGNE.
     PERFORM PTEST-NUMERIQUE.
     PERFORM PTEST-POINT.

  MOVE-A-ARTICLE-VAR.
* ------------------ *
     MOVE NOM-T (I) TO NOM-VARIABLE
     MOVE NOM TO NOM-LIGNE-VAR.
     MOVE SIGNE-T (I) TO SIGNE-TI.
     MOVE NUMERO-T (I) TO RESOURCE.
     PERFORM NUMERO-REEL.
     MOVE 0-AUX TO VALEUR-COEFFICIENT.
     IF SYMBOLE = "s"
        MOVE "N" TO RELATION-VAR
        WRITE ARTICLE-OBJECTIF FROM ARTICLE-VAR
     ELSE IF SYMBOLE = ">"
        MOVE "G" TO RELATION-VAR
     ELSE IF SYMBOLE = "<"
        MOVE "L" TO RELATION-VAR
     ELSE IF SYMBOLE = "="
        MOVE "E" TO RELATION-VAR.
     WRITE ARTICLE-VAR.

  PLUS-UNE.
* ------- *
     ADD 1 TO POINTEUR-CHAINE.

  TYPE-OPTIMISATION.
* ----------------- *
     UNSTRING RECORD-TEST DELIMITED BY ALL " "
              INTO OPTIMISATION
              WITH POINTER POINTEUR-CHAINE.
     IF OPTIMISATION = "MIN" OR = "min"
        MOVE 1 TO ENTIER
        MOVE 0 TO DECI-1
     ELSE IF OPTIMISATION = "MAX" OR = "max"
             MOVE 0 TO ENTIER
             MOVE 0 TO DECI-1
          ELSE DISPLAY
               "Le type d'optimisation doit etre 'MAX' OU 'MIN'"
               DISPLAY "Dans la fonction objectif: " NOM
               MOVE 1 TO SW.
```

```
  TERME-INDEPENDENTE.
* ---------------- *
     IF LIGNE-STRING (POINTEUR-CHAINE) = "+" OR = "-"
        MOVE LIGNE-STRING (POINTEUR-CHAINE) TO SIGNE-TI
        ADD 1 TO POINTEUR-CHAINE
     ELSE MOVE "+" TO SIGNE-TI.
     PERFORM PLUS-UNE UNTIL LIGNE-STRING (POINTEUR-CHAINE) NOT = " ".
     UNSTRING RECORD-TEST DELIMITED BY ALL " "
              INTO RESOURCE COUNT LN
              WITH POINTER POINTEUR-CHAINE.
     MOVE RESOURCE TO NUMERO.
     PERFORM PTEST-NUMERIQUE.
     PERFORM PTEST-POINT.
     PERFORM PTEST-PVERGULE.
     PERFORM PNUMERO-REEL.

*-----------------*
NIVEAU-17 SECTION.
*-----------------*

  TEST-VARIABLE.
* ------------ *
     IF COMPTEUR-VAR > MAXIMUN-VAR
        DISPLAY "On depasse le nombre maximun de variables permi"
        MOVE 1 TO SW.

  PMOUVEMENTS.
* ---------- *
     IF SW = 0
        PERFORM MOUVEMENTS-A-ELEMENTS.

  PTEST-SIGNE.
* ---------- *
     IF SW = 0 AND COMPTEUR-VAR > 1
        PERFORM TEST-SIGNE.

  PTEST-NUMERIQUE.
* -------------- *
     IF SW = 0
        PERFORM TEST-NUMERIQUE VARYING I FROM 1 BY 1
                                       UNTIL I > LN OR SW = 1.

  PTEST-POINT.
* ---------- *
     IF SW = 0
        PERFORM TEST-POINT.

  PTEST-PVERGULE.
* ------------- *
     IF SW = 0
        PERFORM TEST-PVERGULE.

  PNUMERO-REEL.
* ----------- *
     IF SW = 0
        PERFORM NUMERO-REEL.
```

```
*----------------*
NIVEAU-18 SECTION.
*----------------*

  MOUVEMENTS-A-ELEMENTS.
* -------------------- *
     IF COMPTEUR-VAR = 1
        PERFORM TEST-SIGNE-I-ELEMENT
     ELSE MOVE LIGNE-STRING (POINTEUR-CHAINE)
               TO SIGNE-T (COMPTEUR-VAR)
          PERFORM PLUS-UNE.
     PERFORM PLUS-UNE UNTIL
             LIGNE-STRING (POINTEUR-CHAINE) NOT = " ".
     UNSTRING RECORD-TEST DELIMITED BY ALL " "
              INTO NUMERO-T (COMPTEUR-VAR) COUNT LN
                   NOM-T (COMPTEUR-VAR)
              WITH POINTER POINTEUR-CHAINE.
              MOVE NUMERO-T (COMPTEUR-VAR) TO NUMERO.

  TEST-SIGNE.
* --------- *
     IF SIGNE-T (COMPTEUR-VAR) NOT = "+" AND NOT = "-"
        DISPLAY "Il faut un blanc entre nom et signe dans l'equation:"
                NOM
        MOVE 1 TO SW.

  TEST-NUMERIQUE.
* -------------- *
     IF (NUMERO-TEST (I) IS NOT NUMERIC AND NUMERO-TEST (I) NOT = ".")
        MOVE 1 TO SW
        DISPLAY "Il y'a un coeficient non-numerique dans l'equation: "
                NOM.

  TEST-POINT.
* --------- *
     SET K3 TO 1.
     SEARCH NUMERO-TEST VARYING K3 AT END PERFORM MESSAGE-PAS-POINT
            WHEN NUMERO-TEST (K3) = "." PERFORM NUMERO-REEL.

  TEST-PVERGULE.
* ------------ *
     IF LIGNE-STRING (POINTEUR-CHAINE) NOT = ";"
        MOVE 1 TO SW
        DISPLAY "Il manque le "";"" a la fin de l'equation: " NOM.

  NUMERO-REEL.
* ---------- *
     MOVE 1 TO POINTEUR-QUANTITE.
     UNSTRING RESOURCE DELIMITED BY "."
              INTO ENTIER
              WITH POINTER POINTEUR-QUANTITE.
     UNSTRING RESOURCE DELIMITED BY ALL " "
              INTO DECI COUNT L1
              WITH POINTER POINTEUR-QUANTITE.
     COMPUTE DECI-1 = (DECI * 10 ** (5 - L1)) / 10 ** 5.
     IF SIGNE-TI = "-"
        COMPUTE Q-AUX = Q-AUX * (-1).
```

```
*-----------------*
NIVEAU-19 SECTION.
*-----------------*

  TEST-SIGNE-I-ELEMENT.
* ------------------- *
     IF LIGNE-STRING (POINTEUR-CHAINE) = "+" OR = "-"
        MOVE LIGNE-STRING (POINTEUR-CHAINE) TO SIGNE-T (1)
        ADD 1 TO POINTEUR-CHAINE
     ELSE MOVE "+" TO SIGNE-T (1).

  MESSAGE-PAS-POINT.
* ----------------- *
     DISPLAY "Il manque le point dans un coeficient de l'equation:"
             NOM
     MOVE 1 TO SW.
```

```
*++++LECTURE DE DONNES D'APRES L'ECRAN++++++*
LECTURE-DES-ECRAN SECTION.
*++++++++++++++++++++++++++++++++++++++++++*

*
     PERFORM INITIALISATIONS-ECRAN.
*
     PERFORM GROUPEMENT-ECLATEMENT-ECRAN UNTIL FIN-FICHIER = "OUI"
                                                    OR SW = 1.
*
     PERFORM CLOTURES.

*----------------*
NIVEAU-21 SECTION.
*----------------*


  INITIALISATIONS-ECRAN.
* ------------------- *
     DISPLAY " ".
     DISPLAY "Voulez-vous lire les normes pour l'entree de donnees?
-    "(Y ou N):" WITH NO ADVANCING.
     MOVE 0 TO INDICATEUR.
     PERFORM TEST-REPONSE UNTIL INDICATEUR > 0.
     IF REPONSE = "Y" OR = "y"
        PERFORM INDICATIONS.
     PERFORM DEMANDE-EXEMPLE.
     MOVE "NON" TO FIN-FICHIER.
     MOVE 0 TO SW.
     MOVE 1 TO COMPTEUR-EQUATIONS.
     OPEN OUTPUT FICHIER-DATA, FICHIER-LIGNES, FICHIER-FONCTOBJ
                 FICHIER-VAR, FICHIER-OBJECTIFS.
     DISPLAY "$H", "$J" WITH NO ADVANCING.
     PERFORM AVANT-LECTURE-NOM.

  GROUPEMENT-ECLATEMENT-ECRAN.
* ------------------------- *
     MOVE 0 TO ENCORE.
     MOVE "NON" TO BONNE-EQUATION.
     PERFORM EQUATION-VALIDE UNTIL (ENCORE = 3 OR
                                    BONNE-EQUATION = "OUI").
     IF (ENCORE = 3)
        DISPLAY "Vous devez lire attentivement les normes"
        DISPLAY "pour introduire les donnees"
        MOVE 1 TO SW
     ELSE
        PERFORM PECRITURE-FICHIER-LIGNES
        ADD 1 TO COMPTEUR-EQUATIONS
        PERFORM PECRITURE-FICHIER-DATA
        PERFORM PDEMANDE-FIN-DONNEES.
```

```
*----------------*
NIVEAU-22 SECTION.
*----------------*

  INDICATIONS.
* ---------- *
     DISPLAY "$H", "$J" WITH NO ADVANCING.
     DISPLAY "********** Normes Pour Taper Les Donnees **********".
     DISPLAY "1) TAPER UNIQUEMENT LES VARIABLES QUI ONT UN COEFICIENT
-    " DIFFERENT DE ZERO(0)".
     DISPLAY "    IL FAUT EXPLICITER LES COEFFICIENTS UNITAIRES(1)"
     DISPLAY "2) LES NOMS D'EQUATIONS OU DE VARIABLES PEUVENT AVOIR
-    " MAXIMUN 12 CARACTERES".
     DISPLAY "3) TOUT COEFFICIENT, NOM DE VARIABLE ET SIGNE DOIVENT
-    " ETRE PRECEDES".
     DISPLAY "   ET SUIVIS PAR AU MOINS UN(1) CARACTERE BLANC".
     DISPLAY "4) TOUT COEFFICIENT DOIT CONTENIR EXPLICITEMENT LE
-    " POINT DECIMAL(.) ET".
     DISPLAY "   ETRE PRECEDE PAR LE SIGNE + OU - (SAUF POUR LE
-    " TERME INDEPENDENT ET".
     DISPLAY "   AINSI QUE POUR PREMIER COEFFICIENT DE L'EQUATION
-    " ST POSITIF)".
     DISPLAY "5) CHAQUE EQUATION DOIT SE TERMINER (A SA DERNIER
-    " LIGNE) PAR LE".
     DISPLAY "   SYMBOLE POINT-VIRGULE(;) PRECEDE TOUJOURS D'UN
-    " BLANC".
     DISPLAY "6) IL SUFFIT DE TAPER UNE INEGALITE STRICTE( > ou
-    " < ) POUR REMPLACER".
     DISPLAY "   UNE INEGALITE LARGE, DANS LE CAS D'UNE INEQUATION."
     DISPLAY "   PAR CONTRE, EN CE QUI CONCERNE L'EQUATION DE LA
-    " FONCTION OBJECTIF,"
     DISPLAY "   ON TAPERA A LA FIN DE L'EQUATION: $ MAX ; ".
     DISPLAY "                                     ou: $ MIN ; ".
     DISPLAY "   SI ON VEUT MAXIMISER OU MINIMISER RESPECTIVEMENT"
     DISPLAY "7) CHAQUE FOIS QUE VOUS AVEZ TAPE UNE LIGNE, POUSSEZ
-    " SUR LA TOUCHE RETURN".
     DISPLAY "****** Faire Bien Attention Aux Normes S.V.P. *****".
     DISPLAY " ".

  DEMANDE-EXEMPLE.
* ------------- *
     MOVE 0 TO INDICATEUR.
     DISPLAY "Voulez-vous voir un exemple d'entree de donnees?(Y
-    " ou N):" WITH NO ADVANCING.
     PERFORM TEST-REPONSE UNTIL INDICATEUR > 0.
     IF REPONSE = "Y" OR = "y"
        PERFORM LECTURE-FICHIER-EXEMPLE.

  REPETER-INDICATIONS.
* ----------------- *
     DISPLAY " ".
     MOVE 0 TO INDICATEUR.
     DISPLAY "Voulez-vous relire les normes?(Y ou N):"
             WITH NO ADVANCING.
     PERFORM TEST-REPONSE UNTIL INDICATEUR > 0.
     IF REPONSE = "Y" OR = "y"
        PERFORM INDICATIONS.
     PERFORM DEMANDE-EXEMPLE.
```

```
  AVANT-LECTURE-NOM.
* ---------------- *
     MOVE 0 TO ENCORE.
     MOVE "NON" TO BON-NOM.
     PERFORM LECTURE-TEST-NOM-EQUATION UNTIL (ENCORE = 2 OR
                                             BON-NOM = "OUI").

  EQUATION-VALIDE.
* -------------- *
     MOVE 0 TO COMPTEUR-LIGNES.
     MOVE "NON" TO FIN-ARTICLE.
     DISPLAY "Tapez les lignes correspodant a l'equation: ", NOM.
     PERFORM LECTURE-FICH.
     PERFORM GENERATION-ARTICLE-GROUPE-E UNTIL FIN-ARTICLE = "OUT"
                                                OR SW = 1.
     PERFORM PDECOMPOS-ARTICLE-GROUPE-E.
     PERFORM PTEST-NOMBRE-EQUATIONS.
     PERFORM PMESSAGE-RECOMENCER.

  PECRITURE-FICHIER-DATA.
* --------------------- *
     IF SW = 0
        PERFORM ECRITURE-FICHIER-DATA.

  PDEMANDE-FIN-DONNEES.
* ------------------- *
     IF SW = 0
        PERFORM DEMANDE-FIN-DONNEES.

*-----------------*
NIVEAU-23 SECTION.
*-----------------*

  LECTURE-FICHIER-EXEMPLE.
* ---------------------- *
     OPEN INPUT FICHIER-EXEMPLE.
     MOVE "NON" TO  FIN-FICHIER.
     PERFORM LECTURE-EXEMPLE.
     DISPLAY "$H", "$J" WITH NO ADVANCING.
     PERFORM DISPLAY-EXEMPLE UNTIL FIN-FICHIER = "OUI".
     CLOSE FICHIER-EXEMPLE.
     PERFORM REPETER-INDICATIONS.

  TEST-REPONSE.
* ----------- *
     ACCEPT REPONSE.
     IF REPONSE = "Y" OR = "N" OR = "y" OR = "n"
        MOVE 1 TO INDICATEUR
     ELSE DISPLAY "vous devez repondre Y ou N:"
          WITH NO ADVANCING.

  LECTURE-TEST-NOM-EQUATION.
* ------------------------ *
     MOVE " " TO NOM.
     PERFORM LECTURE-NOM-EQUATION UNTIL NOM NOT = " ".
     PERFORM TEST-NOM-LIGNE.
     PERFORM PALARME-NOM-REPETE.
```

```
  GENERATION-ARTICLE-GROUPE-E.
* ------------------------- *
     ADD 1 TO COMPTEUR-LIGNES.
     PERFORM TEST-LIGNES.
     PERFORM PPASSAGE.

  PDECOMPOS-ARTICLE-GROUPE-E.
* ------------------------- *
     IF SW = 0
        PERFORM CORP-DECOMPOSITION-E.

  PMESSAGE-RECOMENCER.
* ------------------- *
     IF (COMPTEUR-VAR > MAXIMUN-VAR OR
         COMPTEUR-LIGNES > MAXIMUN-LIGNES OR
         COMPTEUR-EQUATIONS > MAXIMUN-EQUATIONS)
          MOVE "OUI" TO BONNE-EQUATION
     ELSE PERFORM MESSAGE-RECOMENCER.

  ECRITURE-FICHIER-DATA.
* --------------------- *
     MOVE NOM TO ARTICLE-DATA.
     WRITE ARTICLE-DATA.
     PERFORM CONTINUATION-ECRITURE VARYING I FROM 1 BY 1
                                   UNTIL I > COMPTEUR-LIGNES.

  DEMANDE-FIN-DONNEES.
* ------------------- *
     DISPLAY " "
     MOVE 0 TO INDICATEUR.
     DISPLAY "Vous avez encore de donnees a taper?(Y ou N):"
             WITH NO ADVANCING.
     PERFORM TEST-REPONSE UNTIL INDICATEUR > 0.
     IF REPONSE = "N" OR = "n"
        MOVE "OUI" TO FIN-FICHIER
     ELSE PERFORM AVANT-LECTURE-NOM.
```

```
*----------------*
NIVEAU-24 SECTION.
*----------------*

  LECTURE-EXEMPLE.
* -------------- *
     READ FICHIER-EXEMPLE AT END MOVE "OUI" TO FIN-FICHIER.

  DISPLAY-EXEMPLE.
* -------------- *
     DISPLAY ARTICLE-EXEMPLE.
     PERFORM LECTURE-EXEMPLE.

  LECTURE-NOM-EQUATION.
* ------------------- *
     DISPLAY " ".
     DISPLAY "Tapez le nom de l'equation que vous allez introduire".
     ACCEPT NOM.
     IF NOM = "    "
        DISPLAY "Le nom de l'equation doit avoir au moins un caractere".

  PALARME-NOM-REPETE.
* ----------------- *
     IF (SW = 1 AND ENCORE = 2)
        DISPLAY " "
        DISPLAY "Vous devez recomencer a taper tout avec attention"
        ADD 1 TO ENCORE
     ELSE IF SW = 1
             DISPLAY "Changer le nom de l'equation"
             ADD 1 TO ENCORE
             MOVE 0 TO SW
          ELSE MOVE "OUI" TO BON-NOM.

  CORP-DECOMPOSITION-E.
* ------------------- *
     MOVE 1 TO POINTEUR-CHAINE.
     PERFORM CORPS-DECOMPOSITION.
     PERFORM PSYMBOLE-TI.
     PERFORM PECRITURE-FICHIER-VAR.

  MESSAGE-RECOMENCER.
* ----------------- *
     IF (SW = 1 AND ENCORE = 2)
        DISPLAY "Vous devez recomencer a taper tout avec attention"
        ADD 1 TO ENCORE
     ELSE IF SW = 1
                DISPLAY "Retaper les lignes de l'equation: ", NOM
                ADD 1 TO ENCORE
                MOVE 0 TO SW
             ELSE MOVE "OUI" TO BONNE-EQUATION.

  CONTINUATION-ECRITURE.
* -------------------- *
     WRITE ARTICLE-DATA FROM ITEM (I).
```

```
*========================*
IDENTIFICATION DIVISION.
*========================*

PROGRAM-ID. TICKET.
AUTHOR. AGUILAR VICTOR.

*=============================================================================*
*                                                                             *
* Ce programme permet la lecture du nom du fichier qui contiendra, ou         *
* contient deja, les donnes de la matrice du probleme avec l'extention        *
* DAT; il peut avoir une longueur maximun de 8 caracteres majuscules,         *
* mais ici, on ne tiendra comote que des 6 premiers.                          *
*                                                                             *
* On demande a l'utilisateur quelle sera la facon d'introduire les            *
* les donnees(a partir d'un fichier, ou a partir de l'ecran en mode           *
* interactif).                                                                *
*                                                                             *
*=============================================================================*

*========================*
ENVIRONMENT DIVISION.
*========================*

*--------------------*
INPUT-OUTPUT SECTION.
*--------------------*

FILE-CONTROL.
     SELECT FICHIER-TICKET ASSIGN TO DSK
                    RECORDING MODE IS ASCII.
     SELECT FICHIER-NOM-MATRICE ASSIGN TO DSK
                      RECORDING MODE IS ASCII.

*==============*
DATA DIVISION.
*==============*

*------------*
FILE SECTION.
*------------*

FD   FICHIER-TICKET LABEL RECORD IS STANDARD
                         VALUE ID IS FICHIERD.
01   ARTICLE.
     02 NOM-FICHIER-DATA              PIC X(8)   DISPLAY-7.
     02 FILLER                        PIC X(4)   DISPLAY-7.
     02 CHOIX                         PIC X      DISPLAY-7.
     02 FILLER                        PIC X(7)   DISPLAY-7.

FD   FICHIER-NOM-MATRICE LABEL RECORD IS STANDARD
                        VALUE ID IS "MATRIXNAM".
01   ARTICLE-NOM-M                    PIC X(6)   DISPLAY-7.
```

```
*-----------------------*
WORKING-STORAGE SECTION.
*-----------------------*

01   FICHIERD.
     02 NOM-FICHIER-DIR              PIC X(6)   DISPLAY-7.
     02 EXTENTION                    PIC X(3)   DISPLAY-7
                                                VALUE "MAT".
77   NOM-FICHIER                    PIC X(6)   DISPLAY-7.
77   NOM-MATRICE                    PIC X(6)   DISPLAY-7.
77   INDICATEUR                     PIC 9      COMP.

*====================*
PROCEDURE DIVISION.
*====================*

  SELECTION-ENTREE-DONNEES.
* ------------------------ *
     OPEN INPUT FICHIER-NOM-MATRICE.
     READ FICHIER-NOM-MATRICE AT END
     DISPLAY "ERREUR FIECHIER-NOM-MATRICE".
     MOVE ARTICLE-NOM-M TO NOM-MATRICE, NOM-FICHIER-DIR.
     CLOSE FICHIER-NOM-MATRICE.
     DISPLAY " ".
     DISPLAY " ".
     DISPLAY "TAPEZ LE NOM DU FICHIER QUI CONTIENT (OU QUI CONTIENDRA)".
     DISPLAY "              LES EQUATIONS DU PROBLEME".
     DISPLAY " (Maximum 6 caracteres majuscules)    :" WITH  NO ADVANCING.
     ACCEPT NOM-FICHIER.
     OPEN OUTPUT FICHIER-TICKET.
     DISPLAY " ".
     DISPLAY " ".
     DISPLAY "              MANIERE D'INTRODUIRE LES DONNEES".
     DISPLAY " ".
     DISPLAY "PAR  FICHIER ---> Taper 1 ".
     DISPLAY " ".
     DISPLAY "PAR  ECRAN -----> Taper 2 ".
     MOVE 1 TO INDICATEUR.
     PERFORM TEST-REPONSE UNTIL INDICATEUR = 0.
     MOVE NOM-MATRICE TO NOM-FICHIER-DATA.
     WRITE ARTICLE.
     CLOSE FICHIER-TICKET.
     CALL "DONNES" USING NOM-FICHIER NOM-MATRICE.
     STOP RUN.

  TEST-REPONSE.
* ------------ *
     DISPLAY " ".
     DISPLAY "        Votre choix?:" WITH NO ADVANCING.
     ACCEPT CHOIX.
     DISPLAY " "
     IF CHOIX = "1" OR CHOIX = "2"
        MOVE 0 TO INDICATEUR
     ELSE DISPLAY "Votre reponse doit etre 1 ou 2".
```

T R A I T E M E N T S

```
C=====================================================================C
C  SPECIFICATIONS DU TYPE DES VARIABLES GLOBALES DU PROGRAMME
C  PLHO, DECLARATIONS COMMON AVEC ETIQUETTE, ET INITIALISATIONS
C=====================================================================C
      CHARACTER*12 NAMCOL(100),NAMFOB(9),NAMEL,NAMEVL,NAMEF,
     *             NAMEVF,NAMAUX,NAMVHB(100),NL(91),LZW(9),
     *             NAMVEC(91),MARQUE(9)*1,TRL*1,ENDFIC*3,
     *             TRF*1,NAMCOT(191),NOMAT*6,FICHD*10,FICHL*10,
     *             FICHV*10,FICHO*10,FICHF*10,MINIMA*3(9)

      REAL*8 TOP(9),POIDS(9),VALVAR(9,100),COEVAR(9,100),
     *       ZVAL,ZCOEF,WVHB(9,100),COEFOB(100),VALFOB(9),
     *       VOVHB(91),VFOVHB(9,100),X(100),CR,COFAUX(100),
     *       COUTR(100),VFOAUX(9),Y(100),Z(100),W(9),MINF(9),
     *       SCOEF2(9),CIBLE(9),TPO(9,9),SOLUT(191)

      INTEGER*4 RDTAUX(100)

      COMMON /CHAR/ NAMCOL,NAMFOB,NAMEL,NAMEVL,NAMAUX,NAMEF,
     *              NAMEVF,TRL,TRF,ENDFIC,LZW,NL,NAMVEC,NAMVHB,
     *              MARQUE,NAMCOT,NOMAT,FICHD,FICHL,FICHV,FICHO,
     *              FICHF,MINIMA

      COMMON /DP/ TOP,VALVAR,COEVAR,ZVAL,WVHB,COEFOB,ZCOEF,
     *            VALFOB,VOVHB,VFOVHB,X,CR,COFAUX,COUTR,
     *            POIDS,VFOAUX,Y,Z,W,CIBLE,TPO,MINF,SCOEF2,
     *            SOLUT

      COMMON /AUTRES/ NOFOB,NOCOL,RDTAUX,KCLEF,KCLEFA,INF,
     *                NVHB,IND,EPSI,IT,T(6),ALFA(9),PI(9),
     *                TOPT(9,6),NUMFO,DDELTA,M,NOCOLT

      DATA T /0.0,0.2,0.4,0.6,0.8,1.0/
      DATA NL /'01','02','03','04','05','06','07','08','09','10',
     *'11','12','13','14','15','16','17','18','19','20','21','22',
     *'23','24','25','26','27','28','29','30','31','32','33','34',
     *'35','36','37','38','39','40','41','42','43','44','45','46',
     *'47','48','49','50','51','52','53','54','55','56','57','58',
     *'59','60','61','62','63','64','65','66','67','68','69','70',
     *'71','72','73','74','75','76','77','78','79','80','81','82',
     *'83','84','85','86','87','88','89','90','91'/
      DATA LZW /'ZW1','ZW2','ZW3','ZW4','ZW5','ZW6','ZW7',
     *          'ZW8','ZW9'/
      DATA RDTAUX /100*0/
      DATA NAMVEC /'VE01','VE02','VE03','VE04','VE05','VE06','VE07',
     *'VE08','VE09','VE10','VE11','VE12','VE13','VE14','VE15','VE16',
     *'VE17','VE18','VE19','VE20','VE21','VE22','VE23','VE24','VE25',
     *'VE26','VE27','VE28','VE29','VE30','VE31','VE32','VE33','VE34',
     *'VE35','VE36','VE37','VE38','VE39','VE40','VE41','VE42','VE43',
     *'VE44','VE45','VE46','VE47','VE48','VE49','VE50','VE51','VE52',
     *'VE53','VE54','VE55','VE56','VE57','VE58','VE59','VE60','VE61',
     *'VE62','VE63','VE64','VE65','VE66','VE67','VE68','VE69','VE70',
     *'VE71','VE72','VE73','VE74','VE75','VE76','VE77','VE78','VE79',
     *'VE80','VE81','VE82','VE83','VE84','VE85','VE86','VE87','VE88',
     *'VE89','VE90','VE91'/
```

```
C*****************************************************************

      PROGRAM PLMO

C*****************************************************************

      INCLUDE 'COMPL.MOB/NOLIST'
C
C
C                     DEFINITION DE VARIABLES GLOBALES
C                     ================================
C
C FICHD: Nom du fichier contenant le nom de la matrice de donnees
C        du probleme, qui est le meme pour le nom de la directorie
C FICHL: Nom du fichier contenant pour chaque equation du probleme:
C        le nom, le type de relation, et, la valeur du terme inde-
C        pendant(RHS)
C FICHV: Nom du fichier contenant l'information de chaque terme de
C        la matrice du probleme(nom de l'equation, nom de la variable,
C        et, la valeur du coefficient)
C FICHF: Nom du fichier contenant pour chaque fonction objectif du
C        probleme, son nom et le type d'optimisation(1 pour minimi-
C        sation, 0 pour maximisation)
C FICHO: Nom du fichier contenant l'information de chaque terme pour
C        chacune des fonctions objectifs(nom de l'objectif, nom de la
C        variable, valeur du coefficient)
C NAMEL,NAMEF: Nom d'une equation
C NAMEVL,NAMEVF: Nom d'une variable
C NAMFOB: Vecteur contenant les noms des fonctions objectifs
C NAMCOL: Vecteur contenant les noms des variables
C NAMAUX: Nom auxiliaire d'une variable ou d'une equation
C MINIMA: Contient le signe (1) pou indiquer qu'un objectif est
C         a minimiser
C TRL,TRF: Type de relation d'une equation: peuvent avoir les valeurs:
C                'G' par une inegalite >=
C                'L' par une inegalite <=
C                'E' par une egalite   =
C                'N' par une fonction objectif
C ENDFIC: Indicateur de fin de fichier
C VALVAR: Matrice contenant les valeurs des solutions pour chaque
C         objectif optimise individuellement
C COEVAR: Matrice contenant les valeurs des coefficients des variables
C         pour chaque fonction objectif
C VALFOB: Vecteur contenant les valeurs des fonctions objectifs par
C         une solution donnee
C X: Vecteur contenant les valeurs d'une solution efficace
C COEFOB: Vecteur contenant les valeurs des coefficients des variables
C         pour une fonction objectif composee(global ou unique)
C ZVAL,ZCOEF: Valeur d'un coefficient
C TOP: Vecteur contenant les types d'optimisation (0 pour maximisation,
C      1 pour minimisation) pour chaque objectif
C NOFOB: Nombre de fonctions objectifs du prooleme
C NOCOL: Nombre de variables de decision du probleme
C INF: Indique si un programme lineaire a une solution optimale
C      ( 0: solution optimale; > 0: pas de solution optimale)
C Fichiers de travail:
C          FOR24.DAT = contient les resultats de la resolution
C                      d'un probleme de PLS
C          FOR31.DAT, FOR41.DAT = contient les informations sur
C                                 les lignes de la matrice
```

```
C          FOR32.DAT, FOR42.DAT = contient les informations sur
C                                 les termes de la matrice
C          FOR33.DAT = contient le(s) noms des objectifs du
C                      probleme et le type d'optimisation
C          FOR35.DAT = contient les informations sur les
C                      termes des fonctions objectifs
C
C                    ============================

C Entree de donnees du probleme
      CALL ENTREE(IFED)
      IF (IFED .NE. 'OK') GO TO 1000
C Coordinateur chargee de l'execution et/ou
C actualisation de donnees du probleme
      CALL GERANT
C Cloture de fichiers
      CALL FINIR
1000  STOP
      END
```

```
C**********************************C
      SUBROUTINE ENTREE(IFED)
C**********************************C

      INCLUDE 'COMPL.MOB/NOLIST'
C
C               DEFINITION DE VARIABLES LOCALES
C               ===============================
C
C IFED: Indicateur si la saisie de donnees c'est bien deroulee:
C       ( s'il existe des erreurs ou pas )
C NOMAT: Nom de la matrice du probleme: ce nom est assigne
C        aux fichiers contenant les donnees en sequence
C
C                       ========================

      CHARACTER*1 BLANC,NOMF*7
      DATA BLANC/' '/
C Demande du nom de la matrice a l'utilisateur
      WRITE(5,2000)
      READ(5,3000)NOMAT
      NOMF = NOMAT
C Assignation du nom aux fichiers contenant les donnees en sequence
      FICHD = NOMAT
      FICHL = NOMAT
      FICHV = NOMAT
      FICHO = NOMAT
      FICHF = NOMAT
      I = INDEX(NOMF,BLANC)
      FICHD(I:I+3) = '.MAT'
      FICHL(I:I+3) = '.LIG'
      FICHV(I:I+3) = '.VAR'
      FICHO(I:I+3) = '.OBJ'
      FICHF(I:I+3) = '.FOB'
C Determiner s'il existent deja les donnees en sequence pour
C le probleme a resoudre
      OPEN(20,FILE=FICHD,ACCESS='SEQIN',ERR=10)
      CLOSE(20)
      IFED = 'OK'
      GO TO 1000
10    OPEN(30,FILE='MATRIX.NAM',ACCESS='SEQOUT')
      WRITE(30,3000)NOMAT
      CLOSE(30)
C Appel a la sous-routine chargee de realiser la saisie de donnees
      CALL CREPRO('TICKET',6,IERR)
      OPEN(20,FILE=FICHD,ACCESS='SEQIN')
      READ(20,4000)IFED
      CLOSE(20)
1000  RETURN
2000  FORMAT('1','TAPEZ LE NOM DE LA MATRICE DU PROBLEME'/,
     1 '  (Maximum 6 caracteres majuscules)    :',$)
3000  FORMAT(A6)
4000  FORMAT(13X,A2)
      END
```

```
C*****************************$
      SUBROUTINE GERANT
C*****************************$

      INCLUDE 'COMPL.MOB/NOLIST'
C
C Selection du type de traitement par l'utilisateur
C
      R = '5'
      DO WHILE (R .NE. '3')
         WRITE(5,2000)
10       READ(5,3000)R
         IF (R .EQ. '1') THEN
C Appel au module pour l'execution du probleme
            CALL RUNPLM
         ELSE IF (R .EQ. '2') THEN
C Appel au module pour l'actualisation de la matrice
            CALL MAJ
         ELSE IF (R .NE. '3') THEN
            WRITE(5,4000)
            GO TO 10
         END IF
      END DO
      RETURN
2000  FORMAT('1','UTILISATION DU LOGICIEL DE PROGRAMATION',
     1 ' LINEAIRE MULTI-OBJECTIFS'/1X,64('=')//1X,'POUR RESOU',
     2 'DRE LE PROBLEME ----------------> Taper 1'//1X,'POUR ',
     3 'ACTUALISER LA MATRICE DE DONNEES ---> Taper 2'//1X,
     4 'POUR FINALISER ---------------------------> Taper 3'//20X,
     5 'Votre choix:',$)
3000  FORMAT(A1)
4000  FORMAT(1X,'Vous devez taper 1, 2, ou 3:',$)
      END
```

```
C*******************************C
      SUBROUTINE RUNPLM
C*******************************C

      INCLUDE 'COMPL.MOB/NOLIST'
C
C Selection de la methode a utiliser pour
C l'execution du probleme
C
      WRITE(5,1000)
10    WRITE(5,2000)
      READ(5,3000)R
      IF (R .EQ. '1') THEN
C Execution par la methode du Critere global
         CALL CRIGLO
      ELSE IF (R .EQ. '2') THEN
C Execution par la methode Stem
         CALL STEM
      ELSE IF (R .EQ. '3') THEN
C Execution par la methode de Zionts et Wallenius
         CALL ZIONTS
C Effacer les fichiers de travail particulieres
C pour la methode de Zionts et Wallenius
         CALL EFFORZ
      ELSE IF (R .EQ. '4') THEN
C Execution par la methode de Geoffrion
         CALL GEOFRI
      ELSE IF (R .EQ. '5') THEN
C Execution par la methode Goal programming interactif
         CALL GOALP
      ELSE
         WRITE(5,4000)
         GO TO 10
      END IF
      RETURN
1000  FORMAT('1','METHODES POUR RESOUDRE UN PROBLEME DE',
     1 ' PROGRAMATION LINEAIRE MULTIOBJECTIFS'/1X,75('-'),
     2 /1X,'1: Critere Global'/1X,'2: STEM'/1X,'3: Zionts',
     3 ' et Wallenius'/1X,'4: Geoffrion'/1X,'5: Goal Progra',
     4 'mming (Si vous disposez des valeurs cibles pour les',
     5 ' objectifs)')
2000  FORMAT(/10X,'Votre Choix:',$)
3000  FORMAT(A1)
4000  FORMAT(1X,'Vous devez taper: 1,2,3,4, ou 5')
      END
```

```
C==============================C
      SUBROUTINE SORTIE
C==============================C
C
C Parametres(NOFOB,NOCOL,TOP,NAMFOB,NAMCOL,VALFOB,X)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Sortie de resultats d'une solution (efficace ou possible)
C
      R = 'Y'
      IMP = 5
      K = 1
      DO WHILE((R .EQ. 'Y' .OR. R .EQ. 'y') .AND. K .LE. 2)
C Affichage de la solution sur l'ecran ou impression sur papier
         WRITE(IMP,1000)
         DO L=1,NOFOB
            I = L
            WRITE(IMP,2000)NAMFOB(I),VIMP(VALFOB(I),I,TOP(I))
         END DO
         WRITE(IMP,3000)(NAMCOL(J),X(J),J=1,NOCOL)
         IF (K .EQ. 1) THEN
            WRITE(5,4000)
            CALL TESTR(R)
            IF (R .EQ. 'Y' .OR. R .EQ. 'y') IMP = 3
         END IF
         K = K + 1
      END DO
      RETURN
1000  FORMAT('1'//15X,'SOLUTION EFFICACE'/15X,17('=')//10X,
     1'OBJECTIF',10X,'VALEUR'/10X,8('-'),10X,6('-'))
2000  FORMAT(10X,A12,4X,SP,E14.7)
3000  FORMAT(//10X,'ACTIVITE',10X,'VALEUR'/10X,8('-'),10X,
     1 6('-'), /(10X,A12,4X,SP,E14.7))
4000  FORMAT(/1X,'Voulez-Vous avoir ces resultats sur',
     1 ' l''imprimante?(Y ou N):',$)
      END

C------------------------------C
      SUBROUTINE FACON
C------------------------------C

      CHARACTER*79 LINREC
C Lecture d'un fichier contenant un probleme-exemple
10    CONTINUE
      READ(29,2000,END=1000)LINREC
      WRITE(5,*)LINREC
      GO TO 10
1000  RETURN
2000  FORMAT(A79)
      END
```

```
C-----------------------------------C
      FUNCTION VIMP(V,I,T)
C-----------------------------------C

      REAL*8 V,VIMP,T
C
C             DEFINITION DE VARIABLES LOCALES
C             ===============================
C V: Valeur de la fonction objectif
C I: Numero de la fonction objectif dans la sequence
C    du probleme
C T: Type d'optimisation de la fonction objectif
C

C Si le type d'otimisation est minimisation, il faut changer
C la valeur de la fonction objectif avant de l'imprimer
      IF (T .EQ. 1.D0) THEN
         VIMP = -V
      ELSE
         VIMP = V
      END IF
      RETURN
      END

C-----------------------------------C
      SUBROUTINE TESTR(R)
C-----------------------------------C

C
C Test si la reponse de l'utilisateur est "y" ou "n"
C
      READ(5,2000)R
      DO WHILE(R .NE. 'Y' .AND. R .NE. 'y' .AND.
     1         R .NE. 'N' .AND. R .NE. 'n')
         WRITE(5,1000)
         READ(5,2000)R
      END DO
      RETURN
1000  FORMAT(1X,'Votre reponse doit etre Y ou N:',$)
2000  FORMAT(A1)
      END
```

```
C-----------------------------------------------C
      SUBROUTINE LECL1(N,N1,TR,VAL,ENDFIC)
C-----------------------------------------------C

      CHARACTER*12 N1,TR*1,ENDFIC*3
      REAL*8 VAL
C
C             DEFINITION DE VARIABLES LOCALES
C             ===============================
C N: Numero du fichier a lire
C N1: Nom de l'equation
C TR: Type de ralation
C VAL: Valeur du terme
C ENDFIC: Indicateur de fin de fichier
C
      READ(N,2000,END=1000)N1,TR,VAL
      RETURN
1000  ENDFIC = 'OUT'
      RETURN
2000  FORMAT(A12,A1,F15.5)
      END

C-----------------------------------------------C
      SUBROUTINE LECV1(N,N1,N2,TR,VAL,ENDFIC)
C-----------------------------------------------C

      CHARACTER*12 N1,N2,TR*1,ENDFIC*3
      REAL*8 VAL
C
C             DEFINITION DE VARIABLES LOCALES
C             ===============================
C N: Numero du fichier a lire
C N1: Nom de l'equation
C N2: Nom de la variable
C TR: Type de ralation
C VAL: Valeur du terme
C ENDFIC: Indicateur de fin de fichier
C
      READ(N,2000,END=1000)N1,N2,TR,VAL
      RETURN
1000  ENDFIC = 'OUT'
      RETURN
2000  FORMAT(2A12,A1,F15.5)
      END
```

```
C-------------------------------------------------------C
      SUBROUTINE LECV2(N,N1,N2,TR,VAL,CR,ENDFIC)
C-------------------------------------------------------C

      CHARACTER*12 N1,N2,TR*1,ENDFIC*3
      REAL*8 VAL,CR
C
C
C             DEFINITION DE VARIABLES LOCALES
C             ===============================
C N: Numero du fichier a lire
C N1: Nom de l'equation
C N2: Nom de la variable
C TR: Type de relation
C VAL: Valeur du terme
C CR: Valeur du cout relative de la variable
C ENDFIC: Indicateur de fin de fichier
C
      READ(N,2000,END=1000)N1,N2,TR,VAL,CR
      RETURN
1000  ENDFIC = 'OUI'
      RETURN
2000  FORMAT(2A12,A1,2F15.5)
      END

C-----------------------------------C
      SUBROUTINE EFFOR2
C-----------------------------------C

C Effacer les fichiers de travail crees
C par la methode de Zionts-Wallenius

      OPEN(24,ACCESS='SEQOUT')
      OPEN(36,ACCESS='SEQOUT')
      OPEN(37,ACCESS='SEQOUT')
      OPEN(38,ACCESS='SEQOUT')
      CLOSE(24,DISPOSE='EXPUNGE')
      CLOSE(36,DISPOSE='EXPUNGE')
      CLOSE(37,DISPOSE='EXPUNGE')
      CLOSE(38,DISPOSE='EXPUNGE')
      RETURN
      END
```

```
C*************************************************************
      SUBROUTINE STEM
C*************************************************************

      INCLUDE 'COMPL.HOB/NOLIST'
C
C
C                  DEFINITION DE VARIABLES LOCALES
C                  ===============================
C
C NL: Vecteur contenant des noms pour les equations a ajouter
C     a l'ensemble de contraintes
C MARQUE: Vecteur contenant le symbole '*' pour indiquer les
C         objectifs deja relaches
C TPO: Table des pay-offs
C MINF: Vecteur contenant les valeurs minimales pour chaque
C       colonne de la table  pay-off
C SCOEF2: Vecteur contenant la somme des carres des coefficients
C         pour chaque fonction objectif
C PI: Vecteur dont les valeurs sont les poids qu'indique l'importance
C     relative des distances a la solution ideale
C ALFA: Vecteur permettant realiser la ponderation des elements
C       du vecteur PI
C SALFA: Somme des elements du vecteur ALFA
C NUMFO: Numero de l'objectif que le decideur veut relacher pour
C        avoir une nouvelle solution de compromis
C DELTA: Valeur de relachement d'un objectif
C M: Valeur qui permet de generer les noms des equations a ajouter
C    a l'ensemble de contraintes
C
C                     ====================
C

C Demande a l'utilisateur s'il veut regarder un probleme-exemple
      CALL VOIPST
C Calcul des solutions ideales
      CALL IDEAL
      IF (INF .GT. 0) GO TO 1000
C Lecture des solutions initiales et initialisations
      CALL INITAB
C Calcul de la table pay-off
      CALL CALTPO
C Iterations de la methode
      DO I=1,NOFOB
         M=I
C Calcul de la solution efficace qui minimise la distance
C vers la solution ideale
         CALL EUSTEM
         IF (INF .GT. 0) GO TO 1000
C Lecture de la solution efficace qui minimise la distance
         CALL LECSOL
C Impression des resultats
         CALL SORTIE
         IF (I .EQ. NOFOB) GO TO 1000
```

```
C Demande pour continuer les iterations
         CALL TEST(R)
         IF ((R .EQ. 'Y') .OR. (R .EQ. 'v')) THEN
C Actualisation de l'ensemble de contraintes
            CALL ACSTEM
         ELSE
            GO TO 1000
         END IF
      END DO
1000  RETURN
      END

C================================C
      SUBROUTINE VOIRST
C================================C

      INCLUDE 'COMPL.MOB/NOLIST'

      WRITE(5,1000)
      CALL TESTR(R)
      IF (R .EQ. 'Y' .OR. R .EQ. 'v') THEN
         OPEN(29,FILE='STEM.REC',ACCESS='SEQIN')
         CALL FACON
         CLOSE(29)
      END IF
      RETURN
1000  FORMAT('1','Voulez-vous regarder un exemple de resolution',
     1 ' d''un probleme'/1X,'avec la methode que vous venez de',
     2 ' choisir?(Y ou N):',$)
      END

C============================C
      SUBROUTINE IDEAL
C============================C

      INCLUDE 'COMPL.MOB/NOLIST'
C
C Passage des donnees de fichiers d'entree aux fichiers
C de travail
      CALL FTRAVA
C Calcul des solutions ideales pour chaque fonction
C objectif pris separement
      CALL OPPLS(FICHD,INF,NOPOB,NOCOL,41,42,0)
      IF (INF .GT. 0) WRITE(5,1000)
      RETURN
1000  FORMAT(//1X,'Les contraintes du probleme multi-objectifs',
     *' ne sont pas bien concues')
      END
```

```
C================================C
      SUBROUTINE INITAB
C================================C
C
C Parametres(VALFOR,TPO,VALVAR,NAMCOL,NOFOB,
C             NOCOL,SCOEF2,MINIMA,COEVA,MAROUE)
C
      INCLUDE 'COMPL.MOD/NOLIST'
C
C Lecture des solutions ideales
C
      OPEN(24,ACCESS='SEQIN')
      KF = 0
      I = 0
      ENDFIC='NON'
      CALL DECV1(24,NAMEL,NAMEVL,IRL,ZVAL,ENDFIC)
      DO WHILE (ENDFIC .EQ.'NON')
         KF = KF + 1
         VALFOR(KF) = ZVAL
         IF (TOP(KF) .EQ. 1.D0) THEN
            TPO(KF,KF) = -ZVAL
            MINIMA(KF) = '(!)'
         ELSE
            TPO(KF,KF) = ZVAL
            MINIMA(KF) = ' '
         END IF
         NAMAUX = NAMEL
         KV = 0
         CALL DECV1(24,NAMEL,NAMEVL,IRL,ZVAL)
         DO WHILE ((NAMEL .EQ. NAMAUX) .AND. (ENDFIC .EQ.'NON'))
            KV = KV + 1
            VALVAR(KF,KV) = ZVAL
            IF (I .EQ. 0) THEN
               NAMCOL(KV) = NAMEVL
            END IF
            CALL DECV1(24,NAMEL,NAMEVL,IRL,ZVAL,ENDFIC)
         END DO
         I = 1
      END DO
      CLOSE(24,DISPOSE='EXPUNGE')
      DO I=1,NOFOB
         DO J=1,NOCOL
            COEVAR(I,J) = 0.D0
         END DO
      END DO
```

```
C Lecture des coefficients des variables dans
C chacune des fonctions objectifs
      OPEN(25,FILE=FICHO,ACCESS='SEQIN')
      ENDFIC = 'NON'
      KF = 0
      CALL LFCV1(25,NAMEF,NAMEVF,TPF,ZCOEF,ENDFIC)
      DO WHILE (ENDFIC .EQ. 'NON')
         KF = KF + 1
         NAMAUX = NAMEF
         SCOEF2(KF) = 0
         KV = 1
         DO WHILE ((NAMAUX .EQ. NAMEF) .AND. (ENDFIC .EQ. 'NON'))
            SCOEF2(KF) = SCOEF2(KF) + ZCOEF**2
            DO WHILE (NAMEVF .NE. NAMCOL(KV))
               KV = KV + 1
            END DO
C Si le type d'optimisation est minimisation
C on change a maximisation(fois -1 la f.o.)
            IF (TOP(KF) .EQ. 1.D0) THEN
               COEVAR(KF,KV) = -ZCOEF
            ELSE
               COEVAR(KF,KV) = ZCOEF
            END IF
            CALL LFCV1(25,NAMEF,NAMEVF,TPF,ZCOEF,ENDFIC)
         END DO
      END DO
      CLOSE(25)
C Initiation de marques qui permettent de signaler les objectifs
C deja relaches
      DO I=1,NOFOB
         MARQUE(I) = ' '
      END DO
      RETURN
      END
```

```
C================================C
      SUBROUTINE CALTPO
C================================C
C
C Parametres(NOFOB,NOCOL,NAMFOB,NAMCOL,TPO,
C             VALVAR,COEVAR,SCOEF2,ALFA)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C
C Impresion des solutions ideales si on le desire
      WRITE(5,1000)
      CALL TESTR(R)
      IF (R .EQ. 'Y' .OR. R .EQ. 'y') THEN
         DO I=1,NOFOB
            T = I
            WRITE(5,3000)NAMFOB(I),VIMP(TPO(I,I),I,TOP(I))
            WRITE(5,4000)(NAMCOL(J),VALVAR(I,J),J=1,NOCOL)
         END DO
      END IF
C Calcul de la table des pay-off
      DO I=1,NOFOB
         DO J=1,NOFOB
            IF (I .NE. J) THEN
               TPO(I,J) = 0
               DO K=1,NOCOL
                  TPO(I,J) = TPO(I,J) + VALVAR(I,K) * COEVAR(J,K)
               END DO
            END IF
         END DO
      END DO
C Calcul du minimun pour chaque colonne de la table pay-off
      DO J=1,NOFOB
         MINF(J) = TPO(1,J)
         DO I=2,NOFOB
            IF (TPO(I,J) .LT. MINF(J)) THEN
               MINF(J) = TPO(I,J)
            END IF
         END DO
      END DO
C Calcul des poids pour les indicateurs de l'importance
C relative des distances
      DO I=1,NOFOB
         IF (TPO(I,I) .GT. 0) THEN
            ALFA(I) = ((TPO(I,I) - MINF(I))/TPO(I,I)) *
     *                 1. / SQRT(SCOEF2(I))
         ELSE
            ALFA(I) = ((MINF(I) - TPO(I,I))/MINF(I)) *
     *                 1. / SQRT(SCOEF2(I))
         END IF
      END DO
      RETURN
1000  FORMAT(/1X,'Voulez-vous regarder les solutions ideales',
     1 ' ces objectifs?(Y ou N):',$)
3000  FORMAT('1',10X,'OBJECTIF',10X,'SOL. IDEALE'/11X,8('='),
     1 10X,11('=')/(11X,A12,6X,SP,E12.5))
4000  FORMAT(/11X,'ACTIVITE',12X,'VALEUR'/11X,8('-'),12X,6('-')
     1 /(11X,A12,6X,SP,E12.5))
      END
```

```
C================================C
      SUBROUTINE FOSTEM
C================================C
C
C Parametres(ALFA,NOFOB,NOCOL,PI,TPO,COEVAR,NAMCOL)
C
      INCLUDE 'COMPO.NOB/NOLIST'

      CHARACTER*12 LIGAUX(9)
      DATA LIGAUX /'STEM1','STEM2','STEM3','STEM4','STEM5',
     *             'STEM6','STEM7','STEM8','STEM9'/
C Calcul des indicateurs de l'importance relative des distances
C vers la solution ideale
      SALFA = 0
      DO I=1,NOFOB
         SALFA = SALFA + ALFA(I)
      END DO
      DO I=1,NOFOB
         PI(I) = ALFA(I) / SALFA
      END DO
C Fonction objective pour minimiser les distances
      OPEN(33,ACCESS='SEQOUT',CARRIAGECONTROL='LIST')
      NAMEF = 'DISTANCESTEM'
      NAMEVF = 'LAMBDASTEM'
      TRF = 'H'
      ZCOEF = 1.D0
      WRITE(33,1000)NAMEF,TRF,ZCOEF
      CLOSE(33)
      OPEN(41,ACCESS='APPEND',CARRIAGECONTROL='LIST')
      OPEN(42,ACCESS='APPEND',CARRIAGECONTROL='LIST')
      TRL = 'G'
      WRITE(42,2000)NAMEF,NAMEVF,TRF,ZCOEF
      DO I=1,NOFOB
         IF (PI(I) .NE. 0.D0) THEN
            ZVAL = TPO(I,1) * PI(I)
            WRITE(41,1000)LIGAUX(I),TRL,ZVAL
            WRITE(42,2000)LIGAUX(I),NAMEVF,TRL,ZCOEF
            DO J=1,NOCOL
               ZVAL = COEVAR(I,J) * PI(I)
               WRITE(42,2000)LIGAUX(I),NAMCOL(J),TPO,ZVAL
            END DO
         END IF
      END DO
      CLOSE(41)
      CLOSE(42)
      CALL SORT('SORT/RECORD:40/KEY:13,12/KEY:1,12
     *           FOR42.DAT FOR42.DAT')
C Optimisation de la fonction objective distance
      CALL OPPLS(FTCHD,INF,N3,N4,41,42,0)
      IF (INF .GT. 0) WRITE(5,3000)
      RETURN
1000  FORMAT(A12,A1,F15.5)
2000  FORMAT(2A12,A1,F15.5)
3000  FORMAT(//1X,'La fonction objectif lambda n''a pas',
     1 ' de solution optimale')
      END
```

```
C=================================C
      SUBROUTINE LECSOL
C=================================C
C
C Parametres(X,NOFOB,NOCOL,COEVAR,VALFOB)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C
C Lecture des solutions
      OPEN(24,ACCESS='SEQIN')
      ENDFIC = 'NON'
      KV = 0
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         IF (NAMEVL .NE. 'LAMBDASTEP') THEN
            KV = KV + 1
            X(KV) = ZVAL
         END IF
         CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      END DO
      CLOSE(24,DISPOSE='EXPUNGE')
      DO I=1,NOFOB
         VALFOB(I) = 0.
         DO J=1,NOCOL
            VALFOB(I) = VALFOB(I) + COEVAR(I,J) * X(J)
         END DO
      END DO
      RETURN
      END
```

```
C==================================C
      SUBROUTINE TEST(R)
C==================================C
C
C Parametres(NOFOB,MINTMA,NAMFOB,MARQUE,TPO,VALFOB,NUMFO,DDELTA)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C
C Demande a l'utilisateur s'il veut relacher la valeur
C d'un objectif
      WRITE(5,1000)
      DO L=1,NOFOB
         I = L
          WRITE(5,1010)I,MINTMA(I),NAMFOB(I),VIMP(TPO(I,I),I,TOP(I)),
     1                   VIMP(VALFOB(I),I,TOP(I)),MARQUE(I)
      END DO
      WRITE(5,1015)
      WRITE(5,1020)
      CALL TESTR(R)
      IF ((R .EQ. 'Y') .OR. (R .EQ. 'y')) THEN
         K = 0
         DO WHILE (K .EQ. 0)
            WRITE(5,1040)
            READ(5,1050)NUMFO
            IF (ALFA(NUMFO) .EQ. 0.) THEN
               WRITE(5,1080)
            ELSE
               MARQUE(NUMFO) = '*'
               K = 1
            END IF
         END DO
         WRITE(5,1060)
         READ(5,*)DDELTA
      END IF
990   RETURN
1000  FORMAT('1'/7X,'OBJECTIF',7X,'SOLUT. IDEALE ',3X,'SOLUT.
     * ACTUELE',/7X,8('='),7X,13('='),4X,14('='))
1010  FORMAT(1X,I1,': ',A3,A12,2X,F15.5,2X,F15.5,2X,A1)
1015  FORMAT(/1X,'(!)---> L''objectif dont son nom est precede',
     1 ' par ce signal est a MINIMISER')
1020  FORMAT(/1X,'Voulez-vouz relacher(diminuer) un objectif?
     * (Y ou N):',$)
1040  FORMAT(1X,'Quel objectif?(Tapez le numero correspondant):',$)
1050  FORMAT(I1)
1060  FORMAT(1X,'Quelle quantite?:',$)
1070  FORMAT(F15.5)
1080  FORMAT(1X,'Objectif deja relache(Tapez-en un qui n''ait pas
     * la marque ''*'')')
      END
```

```
C==============================C
      SUBROUTINE ACSTEM
C==============================C
C
C Parametres(N,NOFOB,NOCOL,NUMFO,DDELTA,VALFOB,NL,NAMCOL,COEVAR)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C
C Actualisation des contraintes du probleme
      CHARACTER*1 A(40),B(28)
      OPEN(31,ACCESS='APPEND',CARRIAGECONTROL='LIST')
      OPEN(32,ACCESS='APPEND',CARRIAGECONTROL='LIST')
      K = 9*(N-1)
      TRL = 'G'
      DO I=1,NOFOB
         IF (I .EQ. NUMFO) THEN
            VALFOB(I) = VALFOB(I) - DDELTA
            ALFA(I) = 0.
         END IF
         WRITE(31,3000)NL(I+K),TRL,VALFOB(I)
         DO J=1,NOCOL
            WRITE(32,4000)NL(I+K),NAMCOL(J),TRL,COEVAR(I,J)
         END DO
      END DO
      CLOSE(31)
      CLOSE(32)
      OPEN(31,ACCESS='SEQIN')
      OPEN(41,ACCESS='SEQOUT')
1     CONTINUE
      READ(31,5000,END=1000)B
      WRITE(41,5000)B
      GO TO 1
1000  CLOSE(31)
      CLOSE(41)
      OPEN(32,ACCESS='SEQIN')
      OPEN(42,ACCESS='SEQOUT')
2     CONTINUE
      READ(32,6000,END=2000)A
      WRITE(42,6000)A
      GO TO 2
2000  CLOSE(32)
      CLOSE(42)
      RETURN
3000  FORMAT(A12,A1,F15.5)
4000  FORMAT(2A12,A1,F15.5)
5000  FORMAT(28A1)
6000  FORMAT(40A1)
      END
```

```
C------------------------------C
      SUBROUTINE FTRAVA
C------------------------------C
C
C Parametres(NOFOB,NAMFOB,TOP)
C
      INCLUDE 'COMPL.MOB/NOLIST'

      CHARACTER*1 A(40),B(28)
C
C
C Passage de l'information des fichiers originaux
C vers les fichiers de travail
C
      OPEN(22,FILE=FICHV,ACCESS='SEQIN')
      OPEN(32,ACCESS='SEQOUT',CARRIAGECONTROL='LIST')
      OPEN(42,ACCESS='SEQOUT',CARRIAGECONTROL='LIST')
1     CONTINUE
      READ(22,5000,END=1000)A
      WRITE(32,5000)A
      WRITE(42,5000)A
      GO TO 1
1000  CLOSE(22)
      CLOSE(32)
      CLOSE(42)
      OPEN(21,FILE=FICHL,ACCESS='SEQIN')
      OPEN(31,ACCESS='SEQOUT',CARRIAGECONTROL='LIST')
      OPEN(41,ACCESS='SEQOUT',CARRIAGECONTROL='LIST')
2     CONTINUE
      READ(21,6000,END=2000)B
      WRITE(31,6000)B
      WRITE(41,6000)B
      GO TO 2
2000  CLOSE(21)
      CLOSE(31)
      CLOSE(41)
      OPEN(23,FILE=FICHF,ACCESS='SEQIN')
      OPEN(33,ACCESS='SEQOUT',CARRIAGECONTROL='LIST')
C Lecture de noms des fonctions objectifs
      ENDFIC = 'NON'
      NOFOB = 0
      CALL LECL1(23,NAMEF,TRF,ZCOEF,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         NOFOB = NOFOB + 1
         NAMFOB(NOFOB) = NAMEF
         TOP(NOFOB) = ZCOEF
         WRITE(33,7000)NAMEF,TRF,ZCOEF
         CALL LECL1(23,NAMEF,TRF,ZCOEF,ENDFIC)
      END DO
      CLOSE(23)
      CLOSE(33)
      RETURN
5000  FORMAT(40A1)
6000  FORMAT(28A1)
7000  FORMAT(A12,A1,F15.5)
      END
```

```
C*************************************************************$
      SUBROUTINE GEOFRI
C*************************************************************$

      INCLUDE 'COMPL.MOB/NOLIST'
C
C
C                 DEFINITION DE VARIABLES LOCALES
C                 ===============================
C
C Y: Vecteur solution du probleme dont la fonction objectif c'est
C    une approximation lineaire de la fonction d'utilite en un
C    point donne
C Z: Vecteur qui donne la direction d'amelioration de la fonction
C    d'utilite
C W: Vecteur des taux marginaux de substitution
C VFOAUX: Vecteur auxiliaire pour les valeurs des objectifs
C T: Vecteur contenant des valeurs entre 0 et 1(pas), qui permettent
C    de calculer des solutions pour les objectifs dans l direction
C    d'amelioration de la fonction d'utilite
C TOPT: Matrice contenant les valeurs des objectifs pour chaque
C       valeur de T(pas)
C IT: Indique la valeur de T dont la solution efficace est choisie
C     par le decideur
C DELTA: Vecteur contenant les variations des objectifs qui perme-
C        ttent de calculer les taux marginaux de substitution
C
C                   ====================
C

C Demande a l'utilisateur s'il veut regarder un probleme-exemple
      CALL VOIRGE
C Calcul des solutions ideales pour chaque objectif
      CALL IDEAL
      IF (ILE .GT. 0) GO TO 1000
C Calcul d'une solution initiale en utilisant
C la premiere iteration de la methode STEM
      CALL INITAB
      CALL CALTPO
      CALL FOSTEM
      IF (INF .GT. 0) GO TO 1000
      CALL LECSOL
C Impresion de la solution initiale
      CALL SORTIE
      IT = 10
      R = 'Y'
C Iterations de la methode
      DO WHILE((R .EQ. 'Y' .OR. R .EQ. 'y') .AND. IT .NE. 1)
C Calcul de la direccion d'amelioration
         CALL DIRAME
C Determination du step-size(valeur de T)
         CALL STEPSI(R)
C Impresion des solutions
         IF (IT .NE. 1) CALL SORTIE
      END DO
1000  RETURN
      END
```

```
C====================================C
      SUBROUTINE DIRAME
C====================================C
C
C Parametres(NOFOB,NOCOL,W,COEVAR,COEFOB,X,Y,Z)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Selection d'un objetif de reference different de zero
C
      I = 1
      NOBREF = 1
      DO WHILE(I .LE. NOFOB)
         IF (VALFOB(I) .NE. 0) THEN
            NOBREF = I
            I = NOFOB + 1
         ELSE
            I = I + 1
         END IF
      END DO
C Calcul de poids (taux marginaux de sustitution)
      CALL TMS(NOBREF)
C Calcul de coefficients de la f.o. Y
      DO J=1,NOCOL
         COEFOB(J) = 0.D0
         DO I=1,NOFOB
            COEFOB(J) = COEFOB(J) + W(I) * COEVAR(I,J)
         END DO
      END DO
C Optimisation de la f.o. qui est une approximation de
C la fonction d'utilite(Y)
      CALL OPTIMY
      OPEN(24,ACCESS='SEQIN')
      ENDFIC = 'NON'
      KV = 0
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         KV = KV + 1
         Y(KV) = ZVAL
         CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      END DO
      CLOSE(24,DISPOSE='EXPUNGE')
C Calcul de la direction d'amelioration
      DO I=1,NOCOL
         Z(I) = Y(I) - X(I)
      END DO
      RETURN
      END
```

```
C================================C
      SUBROUTINE VOIRGE
C================================C

      INCLUDE 'COMPL.MOB/NOLIST'
      WRITE(5,1000)
      CALL TESTR(R)
      IF (R .EQ. 'Y' .OR. R .EQ. 'y') THEN
         OPEN(29,FILE='GEOFER.REG',ACCESS='SEQIN')
         CALL FACON
         CLOSE(29)
      END IF
      RETURN
1000  FORMAT('1','Voulez-vous regarder un exemple de resolution',
     1 ' d''un probleme'/1X,'avec la methode que vous venez de',
     2 ' choisir?(Y ou N):',$)
      END

C--------------------------------C
      SUBROUTINE OPTIMY
C--------------------------------C
C
C Parametres(NOCOL,NAMCOL,COEFOB)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Preparation des fichiers pour lamps
      OPEN(33,ACCESS='SEQOUT')
      NAMEF = 'Y'
      TRF = 'N'
      ZCOEF = 0.D0
      WRITE(33,1000)NAMEF,TRF,ZCOEF
      OPEN(22,FILE=FICHV,ACCESS='SEQIN')
      OPEN(32,ACCESS='SEQOUT')
      ENDFIC = 'NON'
      CALL LECV1(22,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         IF (TRL .NE. 'N') THEN
            WRITE(32,2000)NAMEL,NAMEVL,TRL,ZVAL
         END IF
         CALL LECV1(22,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      END DO
      DO I=1,NOCOL
         WRITE(32,2000)NAMEF,NAMCOL(I),TRF,COEFOB(I)
      END DO
      CLOSE(32)
      CALL SORT('SORT/RECORD:40/KEY:13,12/KEY:1,12
     *           FOR32.DAT FOR42.DAT')
C Optimisation de la fonction objectif global
      CALL OPPLS(FICHD,INF,N1,N2,31,42,0)
      IF (INF .GT. 0) WRITE(5,3000)
      RETURN
1000  FORMAT(A12,A1,F15.5)
2000  FORMAT(2A12,A1,F15.5)
3000  FORMAT(//1X,'La fonction global n''a pas de solution',
     *' optimale')
      END
```

```
C================================C
      SUBROUTINE STEPSI(R)
C================================C
C
C Parametres(NOFOB,NOCOL,TOPT,X,T,Z,NAMFOB,TOP,IT)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Calcul des valeurs des fonctions objectifs pour des petits
C variations de distance(step-size) dans la direction
C d'amelioration
      DO I=1,NOFOB
         DO J=1,6
            TOPT(I,J) = 0.
            DO K=1,NOCOL
               TOPT(I,J)=TOPT(I,J) + COEVAR(I,K) *
     1                    (X(K) + T(J) * Z(K))
            END DO
         END DO
      END DO
      WRITE(5,1000)(I,I=1,6)
      DO L=1,NOFOB
         I = L
         WRITE(5,2000)NAMFOB(I),(VTMP(TOPT(I,J),T,TOP(I)),J=1,6)
      END DO
      WRITE(5,3000)
      READ(5,4000)IT
      IF (IT .NE. 1) THEN
         WRITE(5,5000)
         CALL TESTR(R)
      END IF
C Nouvelles valeurs de X et des fonctions objectifs
      DO I=1,NOCOL
         X(I) = X(I) + T(IT) * Z(I)
         VALFOB(I) = TOPT(I,IT)
      END DO
      RETURN
1000  FORMAT('1',78('*')/1X,'* OBJECTIF ',
     1 6('*',4X,I1,5X),'*'/1X,'*',76('-'),'*')
2000  FORMAT(1X,'*',A10,'*',6(F9.2,' *'))
3000  FORMAT(1X,'*',76('-'),'*'//1X,'Quelle solution(vecteur',
     1 ' colonne) vous preferez?'/1x,'tapez le numero de la ',
     2 'colonne correspondante: ',$)
4000  FORMAT(I1)
5000  FORMAT(/1X,'Vous desirez continuer les iterations?(taper',
     1 ' Y)'/1X,'Dans le cas contrire, pour finir, taper',
     2 ' N'/10x,'Votre choix:',$)
      END
```

```
C=====================================C
      SUBROUTINE TMS(NOBREF)
C=====================================C
C
C Parametres(NOFOB,VALFOB,W)
C
      INCLUDE 'COMPL.MOB/NOLIST'

      DIMENSION DELTA(9)
C
C                  DEFINITION DE VARIABLES LOCALES
C                  ===============================
C DELTA: Vecteur contenant les varations initiales
C        assignes aux objectifs
C DELTAI: Valeur inferieur de l'intervalle
C DELTAS: Valeur superieur de l'intervalle
C DELTAM: Valeur milieu de l'intervalle
C                   ================================

C Determination des taux moyens de substitution a partir
C des preferences souhaitees par le decideur
      WRITE(5,1000)NAMFOB(NOBREF)
      DO I=1,NOFOB
         IF (VALFOB(I) .NE. 0) THEN
            A = 0.05 * VALFOB(I)
         ELSE
            A = 0.1 * 0.05
         END IF
         DELTA(I) = ABS(A)
      END DO
      W(NOBREF) = 1.
      I = 1
      DO WHILE ( I .LE. NOFOB)
         IF (I .NE. NOBREF) THEN
            DO K=1,NOFOB
               VFOAUX(K) = VALFOB(K)
            END DO
            VFOAUX(NOBREF) = VALFOB(NOBREF) + DELTA(NOBREF)
            DELTAI = 0.
            DELTAS = DELTA(I)
10          CALL CHOIX(R,I,NOBREF,DELTAS)
            IF (R .EQ. 'B' .OR. R .EQ. 'b') THEN
               DELTAI = DELTAS
               DELTAS = DELTAS * 2.
               GO TO 10
```

```
            ELSE IF (R .EQ. 'A' .OR. R .EQ. 'a') THEN
               DELTAM = (DELTAI + DELTAS) / 2.
20             CALL CHOIX(R,I,NOBREF,DELTAM)
               IF (R .EQ. 'A' .OR. R .EQ. 'a') THEN
                  DELTAS = DELTAM
                  DELTAM = (DELTAI + DELTAS) / 2.
                  GO TO 20
               ELSE IF (R .EQ. 'B' .OR. R .EQ. 'b') THEN
                  DELTAI = DELTAM
                  DELTAM = (DELTAI + DELTAS) / 2.
                  GO TO 20
               ELSE IF (R .EQ. 'I' .OR. R .EQ. 'i') THEN
                  W(I) = DELTA(NOBREF) / DELTAM
               END IF
            ELSE IF (R .EQ. 'I' .OR. R .EQ. 'i') THEN
               W(I) = DELTA(NOBREF) / DELTAS
            END IF
         END IF
         I = I + 1
      END DO
      WRITE(5,2000)(NAMFOB(I),W(I),I=1,NOFOB)
      RETURN
1000  FORMAT('1',15X,'CALCUL DE TAUX DE SUBSTITUTION'/15X,
     1 30('=')/5X,'Objectif de reference: ',A12)
2000  FORMAT('1',5X,'OBJECTIF',8X,'TAUX DE SUBSTITUTION'/6X,
     1 8('-'),8X,19('-')/(6X,A12,10X,F10.3))
      END
```

```
C-----------------------------------------C
      SUPROUTINE CHOIX(R,I,NOBREF,D)
C-----------------------------------------C
C
C Parametres(VALFOB,MINIMA,NAMFOB,TOP)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C R: Reponse du decideur a la demande de preference
C I: Numero de l'objectif
C NOBREF: Numero de l'objectif de reference
C

C Determination des preferences
      VFOAUX(I) = VALFOB(I) - D
      WRITE(5,1000)
      WRITE(5,2000)(MINIMA(K),NAMFOB(K),VIMP(VALFOB(K),K,TOP(K)),
     1              VIMP(VFOAUX(K),K,TOP(K)),K=1,NOFOB)
      WRITE(5,2500)
      WRITE(5,3000)
10    CONTINUE
      WRITE(5,4000)
      READ(5,5000)R
      IF (R .NE. 'A' .AND. R .NE. 'a' .AND. R .NE. 'B' .AND.
     1    R .NE. 'b' .AND. R .NE. 'I' .AND. R .NE. 'i') THEN
         WRITE(5,6000)
         GO TO 10
      END IF
      RETURN
1000  FORMAT(/4X,'OBJECTIF',10X,'SOLUTION A',6X,'SOLUTION B'/
     1 4X,8('='),10X,8('='),1X,'=',6X,9('='),1X,'=')
2000  FORMAT(1X,A3,A12,4X,F12.3,4X,F12.3)
2500  FORMAT(/1X,'(!)---> L''objectif dont son nom est precede',
     1 ' par ce signal est a MINIMISER')
3000  FORMAT(/1X,'Quelle solution(vecteur) preferez vous?',
     1 '(A ou B)'/1x,'Si vous etes indifferent, taper I')
4000  FORMAT(10X,'Votre choix:',$)
5000  FORMAT(A1)
6000  FORMAT(1X,'Vous devez taper: A ou B ou I')
      END
```

```
C*************************************************S
      SUBROUTINE GOALP
C*************************************************S

      INCLUDE 'COMPL.MOB/NOLIST'
C
C
C                 DEFINITION DE VARIABLES LOCALES
C                 ===============================
C
C Y: Vecteur solution pour le programme lineaire qui minimise les
C    ecarts negatifs
C Z: Vecteur qui donne la direction d'amelioration de la fonction
C    des ecarts negatifs
C W: Vecteur contenant les taux marginaux de substitution
C CIBLE: Vecteur des valeurs cibles pour chaque objectif
C DM: Vecteur contenant les noms des variables correspondant aux
C     ecarts negatifs d'un objectif
C
C                    ========================
C

C Demande a l'utilisateur s'il veut regarder un probleme-exemple
      CALL VOIRGP
C Calcul d'une solution initiale
      CALL SIGOAL
      IF (INF .GT. 0) GO TO 1000
C Iterations pour trouver les solutions efficaces
      IT = 10
      R = 'Y'
      DO WHILE((R .EQ. 'Y' .OR. R .EQ. 'y') .AND. IT .NE. 1)
C Determination des poids(tms) pour les objectifs
         CALL POIDSE
         IF (IT .EQ. 1) GO TO 1000
C Minimisation des ecarts negatifs
         CALL OECART
         IF (INF .GT. 0) GO TO 1000
C Lecture des solutions
         CALL SMINE
C Determination du step-size(valeur de T)
         CALL STEPSI(R)
C Impresion des solutions
         IF (IT .NE. 1) CALL SORTIE
      END DO
1000  RETURN
      END
```

```
C================================C
      SUBROUTINE VOIRGP
C================================C

      INCLUDE 'COMPL.MOB/NOLIST'
      WRITE(5,1000)
      CALL TESTR(R)
      IF (R .EQ. 'Y' .OR. R .EQ. 'y') THEN
         OPEN(29,FILE='GOALP.REG',ACCESS='SEQIN')
         CALL FACON
         CLOSE(29)
      END IF
      RETURN
1000  FORMAT('1','Voulez-vous regarder un exemple de resolution',
     1 ' d''un probleme'/1X,'avec la methode que vous venez de',
     2 ' choisir?(Y ou N):',$)
      END
```

```
C==============================C
      SUBROUTINE SIGOAL
C==============================C
C
C Parametres(NOFOB,NAMFOB,CIBLE,X,VALFOB,COEVAR)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Initisations
      CALL INGOAL
C Lecture des cibles
      CALL CIBLES
C Poids(t.m.s.) pour les objectifs
      DO I=1,NOFOB
         W(I) = 1.
      END DO
      OPEN(21,FILE=FICHL,ACCESS='SEQIN')
      OPEN(41,ACCESS='SEQOUT')
      ENDFIC = 'NON'
      CALL LECL1(21,NAMEL,TRL,ZVAL,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         WRITE(41,1000)NAMEL,TRL,ZVAL
         CALL LECL1(21,NAMEL,TRL,ZVAL,ENDFIC)
      END DO
      TRL = 'G'
      DO I=1,NOFOB
         WRITE(41,1000)NAMFOB(I),TRL,CIBLE(I)
      END DO
      CLOSE(21)
      CLOSE(41)
C Optimisation des ecarts negatifs
      CALL OECART
      IF (INF .GT. 0) GO TO 2000
C Lecture de la solution initiale
      OPEN(24,ACCESS='SEQIN')
      ENDFIC = 'NON'
      KV = 1
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         IF (NAMEVL .EQ. NAMCOL(KV)) THEN.
            X(KV) = ZVAL
            KV = KV + 1
         END IF
         CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      END DO
      CLOSE(24,DISPOSE='EXPUNGE')
C Calcul des valeurs des f.o. pour la solution initiale
      DO I=1,NOFOB
         VALFOB(I) = 0.
         DO J=1,NOCOL
            VALFOB(I) = VALFOB(I) + COEVAR(I,J) * X(J)
         END DO
      END DO
      CALL SORTIE
2000  RETURN
1000  FORMAT(A12,A1,F15.5)
      END
```

```
C================================C
      SUBROUTINE INGOAL
C================================C
C
C Parametres(NOFOB,NAMFOB,TOP,NOCOL,NAMCOL,COEVAR,MINIMA)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Lecture de noms des fonctions objectifs
      OPEN(23,FILE=FICFF,ACCESS='SEQIN')
      NOFOB = 0
      ENDFIC = 'NON'
      CALL LECL1(23,NAMEF,TRF,ZCOEF,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         NOFOB = NOFOB + 1
         NAMFOB(NOFOB) = NAMEF
         TOP(NOFOB) = ZCOEF
         CALL LECL1(23,NAMEF,TRF,ZCOEF,ENDFIC)
      END DO
      CLOSE(23)
C Passage du fichier de variables, de 22 a 32
C Lecture de noms de variables
C Calcul du nombre de variables
      OPEN(32,ACCESS='SEQOUT')
      OPEN(22,FILE=FICHV,ACCESS='SEQIN')
      ENDFIC = 'NON'
      NOCOL = 0
      CALL LECV1(22,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      DO WHILE (ENDFIC .EQ. 'NON')
         NAMAUX = NAMEVL
         NOCOL = NOCOL + 1
         NAMCOL(NOCOL) = NAMEVL
         DO WHILE((NAMAUX .EQ. NAMEVL) .AND. (ENDFIC .EQ. 'NON'))
            IF (TRL .NE. 'N') THEN
               WRITE(32,2000)NAMEL,NAMEVL,TRL,ZVAL
            END IF
            CALL LECV1(22,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
         END DO
      END DO
      CLOSE(22)
      CLOSE(32)
```

```
C Lecture des coefficients des fonctions objectifs
      OPEN(25,FILE=FICHO,ACCESS='SEQIN')
      ENDFIC = 'NON'
      KF = 0
      CALL LECV1(25,NAMEF,NAMEVF,TRF,ZCOEF,ENDFIC)
      DO WHILE (ENDFIC .EQ. 'NON')
         KF = KF + 1
         NAMAUX = NAMEF
         KV = 1
         DO WHILE ((NAMAUX .EQ. NAMEF) .AND. (ENDFIC .EQ. 'NON'))
            DO WHILE (NAMEVF .NE. NAMCOL(KV))
               KV = KV + 1
            END DO
C Si le type d'optimisation est minimisation
C on change a maximisation(fois -1 la f.o.)
            IF (TOP(KF) .EQ. 1.DO) THEN
               COEVAR(KF,KV) = -ZCOEF
               MINIMA(KF) = '(!)'
            ELSE
               COEVAR(KF,KV) = ZCOEF
               MINIMA(KF) = ' '
            END IF
            CALL LECV1(25,NAMEF,NAMEVF,TRF,ZCOEF,ENDFIC)
         END DO
      END DO
      CLOSE(25)
      RETURN
2000  FORMAT(2A12,A1,F15.5)
      END



C================================C
      SUBROUTINE CIBLES
C================================C
C
C Parametres(NOFOB,NAMFOB,CIBLE,TOP)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Demande des valeurs cibles
C
      WRITE(5,1000)
      DO I=1,NOFOB
         WRITE(5,3000)NAMFOB(I)
         READ(5,*)CIBLE(I)
         IF (TOP(I) .EQ. 1.DO .AND. CIBLE(I) .GT. 0) THEN
            CIBLE(I) = -CIBLE(I)
         END IF
      END DO
      RETURN
1000  FORMAT('1',9X,'Valeurs des cibles pour chaque objectif',
     1 /10X,39('='))
3000  FORMAT(/1X,'Tapez la cible pour l''objectif ',A12,':',S)
      END
```

```
C==============================C
      SUBROUTINE POIDSE
C==============================C
C
C Parametres(NOFOB,NAMFOB,VALFOB,CIBLE,W)
C
      INCLUDE 'COMPL.MOB/NOLIST'

      DIMENSION POID(9)
C
C Calcul des poids (taux de substitution) des objectifs
C
      WRITE(5,1000)
      CALL TESTR(R)
      IF (R .EQ. 'Y' .OR. R .EQ. 'y') THEN
         DO I=1,NOFOB
            WRITE(5,3000)NAMFOB(I)
            READ(5,*)POID(I)
         END DO
         DO I=1,NOFOB
            W(I) = POID(I) / POID(1)
         END DO
      ELSE
         I = 1
         DO WHILE (I .LE. NOFOB)
            IF (VALFOB(I) .LT. CIBLE(I)) THEN
               CALL TMS(I)
               I = NOFOB + 2
            ELSE
               I = I + 1
            END IF
         END DO
         IF (I .EQ. (NOFOB+1)) THEN
            CALL SORTIE
            TT = 1
         END IF
      END IF
      RETURN
1000  FORMAT('1','Pourriez-vous donner les poids',
     1 ' pour chaque objectif?(Y ou N):',$)
3000  FORMAT(/1X,'Tapez le poids pour l''objectif ',A12,':',$)
      END
```

```
C================================C
      SUBROUTINE OPCART
C================================C
C
C Parametres(NOFOB,NOCOL,NAMFOB,W,NAMCOL,COEVAR,TMF)
C
      INCLUDE 'COMPL.MOB/NOLIST'

      CHARACTER*12 DM(9)
      DATA DM /'DM1','DM2','DM3','DM4','DM5','DM6','DM7',
     *         'DM8','DM9'/
C
C Preparation des fichiers pour lamps
C
      OPEN(33,ACCESS='SEQOUT')
      NAMEF = 'MTHECARTSN'
      TRF = 'N'
      ZCOEF = 0.D0
      WRITE(33,1000)NAMEF,TRF,ZCOEF
      CLOSE(33)
      OPEN(22,FILE=FICHV,ACCESS='SEQIN')
      OPEN(32,ACCESS='SEQOUT')
      ENDFIC = 'NON'
      CALL LECV1(22,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         IF (TRL .NE. 'N') THEN
            WRITE(32,2000)NAMEL,NAMEVL,TRL,ZVAL
         END IF
         CALL LECV1(22,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      END DO
      ZVAL = 1.D0
      TRL = 'G'
      DO I=1,NOFOB
         WRITE(32,2000)NAMEF,DM(I),TRF,(-W(I))
         WRITE(32,2000)NAMFOB(I),DM(I),TRL,ZVAL
         DO J=1,NOCOL
            IF (COEVAR(I,J) .NE. 0.D0) THEN
               WRITE(32,2000)NAMFOB(I),NAMCOL(J),TRL,COEVAR(I,J)
            END IF
         END DO
      END DO
      CLOSE(22)
      CLOSE(32)
      CALL SORT('SORT/RECORD:40/KEY:13,12/KEY:1,12
     *           FOR32.DAT FOR42.DAT')
C Optimisation de la fonction objectif global
      CALL OPPLS(FICHD,INF,N1,N2,41,42,0)
      IF (INF .GT. 0) WRITE(5,3000)
      RETURN
1000  FORMAT(A12,A1,F15.5)
2000  FORMAT(2A12,A1,F15.5)
3000  FORMAT(//1X,'La fonction global n''a pas de solution',
     *' optimale')
      END
```

```
C=====================================C
      SUBROUTINE SMINE
C=====================================C
C
C Parametres(NAMCOL,Y)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Lecture des solutions du probleme de minimisation
C des ecarts negatifs entre les cibles et la solution actuelle
C
      OPEN(24,ACCESS='SEQIN')
      ENDFIC = 'NON'
      KV = 1
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         IF (NAMEVL .EQ. NAMCOL(KV)) THEN
            Y(KV) = ZVAL
            KV = KV + 1
         END IF
         CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      END DO
      CLOSE(24,DISPOSE='EXPUNGE')
C Calcul de la direccion d'amelioration
      DO I=1,NOCOL
         Z(I) = Y(I) - X(I)
      END DO
      RETURN
      END
```

```
C*************************************************************S
      SUBROUTINE CRIGLO
C*************************************************************S

      INCLUDE 'COMPL.MOB/NOLIST'
C
C Triement de fichiers de donnees d'entree
C
C Calcul d'une solution efficace
      CALL SOLINI
      IF (INF .GT. 0) GO TO 1000
C Impression de Resultats
      CALL SORTIE
1000  RETURN
      END


C==============================C
      SUBROUTINE SOLINI
C==============================C
C
C Parametres(INF)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Calcul de solutions ideales
      CALL IDEAL
      IF (INF .GT. 0) GO TO 1000
C Lecture des solutions ideales et des coefficients
C de chaque fonction objectif
      CALL INITAB
C Calcul des coefficients de la fonction objectif global
      CALL CALCUL
C Optimisation de la fonction objectif global
      CALL GLOBAL
C Lecture de la solution efficace
      CALL SOLFIN
1000  RETURN
      END
```

```
C=============================C
      SUBROUTINE CALCUL
C=============================C
C
C Parametres(NOFOB,NOCOL,NAMFOB,NAMCOL,VALFOB
C             VALVAR,TOP,COEFOB,COEVAR)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Calcul des coefficients de la f.o. global
C
      WRITE(5,1000)
      CALL TESTR(R)
      IF (R .EQ. 'Y' .OR. R .EQ. 'y') THEN
         DO L=1,NOFOB
            I = L
            WRITE(5,3000)NAMFOB(I),VIMP(VALFOB(I),I,TOP(I))
            WRITE(5,4000)(NAMCOL(J),VALVAR(I,J),J=1,NOCOL)
         END DO
      END IF
      DO J=1,NOCOL
         COEFOB(J) = 0.D0
         DO L=1,NOFOB
            I = L
            IF (VALFOB(I) .NE. 0.D0) THEN
               COEFOB(J) = COEFOB(J) - COEVAR(I,J) /
     *                     VIMP(VALFOB(I),I,TOP(I))
            END IF
         END DO
      END DO
1000  FORMAT(/1X,'Voulez-vous regarder les solutions Ideales',
     1 ' des objectifs?(Y ou N):',$)
3000  FORMAT('1',10X,'OBJECTIF',10X,'SOL. IDEALE'/11X,8('='),
     1 10X,11('=')/(11X,A12,6X,SP,E12.5))
4000  FORMAT(/11X,'ACTIVITE',12X,'VALEUR'/11X,8('-'),12X,6('-')
     1 /(11X,A12,6X,SP,E12.5))
      RETURN
      END
```

```
C================================C
      SUBROUTINE GLOBAL
C================================C
C
C Parametres(NOCOL,NAMCOL,COEFOB,INF)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Creation des fichiers pour l'optimisation
C de la fonction objectif global
C
      OPEN(33,ACCESS='SEQOUT',CARRIAGECONTROL='LIST')
      NAMEF = 'GLOBAL'
      TRF = 'N'
      ZCOEF = 1.D0
      WRITE(33,1000)NAMEF,TRF,ZCOEF
      CLOSE(33)
      OPEN(32,ACCESS='APPEND',CARRIAGECONTROL='LIST')
      DO K = 1,NOCOL
         WRITE(32,2000)NAMEF,NAMCOL(K),TRF,COEFOB(K)
      END DO
      CLOSE(32)
      CALL SORT('SORT/RECORD:40/KEY:13,12/KEY:1,12
     *           FOR32.DAT FOR42.DAT')
C Optimisation de la fonction objectif global
      CALL OPPLS(FICHD,INF,N1,N2,41,42,0)
      IF (INF .GT. 0) WRITE(5,3000)
      RETURN
1000  FORMAT(A12,A1,F15.5)
2000  FORMAT(2A12,A1,F15.5)
3000  FORMAT(//1X,'La fonction global n''a pas de solution',
     *' optimale')
      END
```

```
C==============================C
      SUBROUTINE SOLFIN
C==============================C
C
C Parametres(X,VALFOB,COEVAR)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Lecture de la solution pour la fonction objective global
C
      OPEN(24,ACCESS='SEQIN')
      ENDFIC = 'NON'
      KV = 0
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         KV = KV + 1
         X(KV) = ZVAL
         CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      END DO
C Calcul des valeurs des fonctions objectifs
      DO I=1,NOFOB
         VALFOB(I) = 0.D0
         DO J=1,NOCOL
            VALFOB(I) = VALFOB(I) + X(J) * COEVAR(I,J)
         END DO
      END DO
      CLOSE(24,DISPOSE='EXPUNGE')
      RETURN
      END
```

```
C*************************************************************$
      SUBROUTINE ZIONTS
C*************************************************************$

      INCLUDE 'COMPL.MOB/NOLIST'
C
C
C                 DEFINITION DE VARIABLES LOCALES
C                 ===============================
C
C NL: Vecteur contenant des noms pour les equations a ajouter a
C     l'ensemble des contraintes
C LZW: Vecteur contenant des noms pour les equations correspondant
C      aux fonctions objectifs a ajouter a l'ensemble de contraintes
C NAMVHB: Vecteur contenant les noms des variables qui sont hors
C         base apres l'optimisation de la fonction d'utilite
C NAMVEC: Vecteur contenant les noms a assigner aux variables d'ecart
C POIDS: Vecteur contenant les poids que l'on donne a chaque fonction
C        objectif dans la fonction d'utilitee
C VOVHB: Vecteur des solutions optimales pour les fonctions corres-
C        pondant aux variables hors base
C VFOVHB: Vecteur des valeurs des objectifs pour les solutions
C         optimales des variables hors base
C COUTR: Cout relative des variables hors-base
C NOCOLT: Nombre de colonnes toales du probleme
C NAMCOLT: Noms de toutes les colonnes du probleme
C SOLUT: Valeurs de toutes les variables dans la solution
C NVHB: Nombre de variables hors-base
C WVHB: Variations des objectifs quand une variable hors-base
C       entre dans la base
C
C                 ===============================
C
```

```
C Demande a l'utilisateur s'il veut regarder un probleme-exemple
      CALL VOIRZW
C Initiations: lecture des donnes, poids initials
      CALL INITZW
      IT = 1
C Iterations de la methode
      DO WHILE((IT .LT. 9) .AND. (KCLEFA .LT. (90 - NVHB)))
C Calcul des coefficients de la fonction d'utilite
         CALL COEFUT
C Generation et optimisation de la fonction d'utilite
         CALL FUTZW
         IF (INF .GT. 0) GO TO 1000
C Lecture des variable hors-base
         CALL VARHB
C Impression des resultats
         CALL SORTIE
         WRITE(5,2000)
         CALL TESTR(R)
         IF (R .EQ. 'N' .OR. R .EQ. 'n') GO TO 1000
C Optimisation de chaque v.h.b.
         CALL OPTVHB
C Selection des v.h.b
         CALL SLVHB
C Determiner les v.h.b. efficaces
         CALL VEFFIC
         IF (KCLEF .EQ. 0) GO TO 1000
C Generation d'une matrice avec les variations choisies
C pour chaque v.h.e.(acceptee ou refussee)
         CALL FVEF
         IF (IND .EQ. 0) GO TO 1000
C Calcul d'une solution possible pour la matrice de variations
         CALL PETRAV
         IF (INF .GT. 0) GO TO 1000
C Lecture de nouveaux poids pour la fonction d'utilitee
         CALL NPOIDS
         IT = IT + 1
      END DO
1000  RETURN
2000  FORMAT(/1X,'Voulez-vous continuer les iterations?(Y ou N):',$)
      END
```

```
C================================C
      SUBROUTINE VOIRZW
C================================C

      INCLUDE 'COMPL.MOB/NOLIST'

      WRITE(5,1000)
      CALL TESTR(R)
      IF (R .EQ. 'Y' .OR. R .EQ. 'y') THEN
         OPEN(29,FILE='ZIUNTS.REG',ACCESS='SEQIN')
         CALL FACON
         CLOSE(29)
      END IF
      RETURN
1000  FORMAT('1','Voulez-vous regarder un exemple de resolution',
     1 ' d''un probleme'/1X,'avec la methode que vous venez de',
     2 ' choisir?(Y ou N):',$)
      END



C================================C
      SUBROUTINE INITZW
C================================C
C
C Parametres(NOFOB,NAMFOB,TOP,NOCOLT,NAMCOT
C             NAMVEC,COEVAR,MINIMA,POIDS)
C
      INCLUDE 'COMPL.MOB/NOLIST'

      EPSI = 1.E-3
      OPEN(23,FILE=FICHF,ACCESS='SEQIN')
C Accumulateur du nombre de variables efficaces, acceptees ou refusees
      KCLEFA = 0
C Lecture de noms des fonctions objectifs
      NOFOB = 0
      ENDFIC = 'NON'
      CALL LECL1(23,NAMEF,TRF,ZCOEF,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         NOFOB = NOFOB + 1
         NAMFOB(NOFOB) = NAMEF
         TOP(NOFOB) = ZCOEF
         CALL LECL1(23,NAMEF,TRF,ZCOEF,ENDFIC)
      END DO
      CLOSE(23)
```

```
C Passage du fichier de variables, de 22 a 32
C Lecture de noms de variables
C Calcul du nombre de variables
      OPEN(32,ACCESS='SEQOUT')
      OPEN(22,FILE=FICHV,ACCESS='SEQIN')
      ENDFIC = 'NON'
      NOCOLT = 0
      CALL LECV1(22,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      DO WHILE (ENDFIC .EQ. 'NON')
         NAMAUX = NAMEVL
         NOCOLT = NOCOLT + 1
         NAMCOT(NOCOLT) = NAMEVL
         DO WHILE((NAMAUX .EQ. NAMEVL) .AND. (ENDFIC .EQ. 'NON'))
            IF (TRL .NE. 'N') THEN
               WRITE(32,2000)NAMEL,NAMEVL,TRL,ZVAL
            END IF
            CALL LECV1(22,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
         END DO
      END DO
      CLOSE(22)
C Passage du fichier de lignes de 21 a 41
      OPEN(21,FILE=FICHL,ACCESS='SEQIN')
      OPEN(41,ACCESS='SEQOUT')
      I = 0
      ENDFIC = 'NON'
      CALL LECL1(21,NAMEL,TRL,ZVAL,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         IF (TRL .EQ. 'G') THEN
            TRL = 'E'
            VAL = -1.D0
            I = I + 1
            NOCOLT = NOCOLT + 1
            NAMCOT(NOCOLT) = NAMVEC(I)
            WRITE(41,1000)NAMEL,TRL,ZVAL
            WRITE(32,2000)NAMEL,NAMVEC(I),TRL,VAL
         ELSE IF (TRL .EQ. 'L') THEN
            TRL = 'E'
            VAL = 1.D0
            I = I + 1
            NOCOLT = NOCOLT + 1
            NAMCOT(NOCOLT) = NAMVEC(I)
            WRITE(41,1000)NAMEL,TRL,ZVAL
            WRITE(32,2000)NAMEL,NAMVEC(I),TRL,VAL
         ELSE
            WRITE(41,1000)NAMEL,TRL,ZVAL
         END IF
         CALL LECL1(21,NAMEL,TRL,ZVAL,ENDFIC)
      END DO
      CLOSE(21)
      CLOSE(41)
      DO I=1,NOFOB
         DO J=1,NOCOLT
            COEVAR(I,J) = 0.D0
         END DO
      END DO
```

```
C Classement des variables
      DO I=1,(NOCOLT - 1)
         DO J=I+1,NOCOLT
            IF (NAMCOT(I) .GT. NAMCOT(J)) THEN
               NAMAUX = NAMCOT(I)
               NAMCOT(I) = NAMCOT(J)
               NAMCOT(J) = NAMAUX
            END IF
         END DO
      END DO
C Lecture des coefficients des fonctions objectifs
      OPEN(25,FILE=FICHO,ACCESS='SEQIN')
      ENDFIC = 'NON'
      KF = 0
      CALL LECV1(25,NAMEF,NAMEVF,TRF,ZCOEF,ENDFIC)
      DO WHILE (ENDFIC .EQ. 'NON')
         KF = KF + 1
         NAMAUX = NAMEF
         KV = 1
         DO WHILE ((NAMAUX .EQ. NAMEF) .AND. (ENDFIC .EQ. 'NON'))
            DO WHILE (NAMEVF .NE. NAMCOT(KV))
               KV = KV + 1
            END DO
C Si le type d'optimisation est minimisation
C on change a maximisation(fois -1 la f.o.)
            IF (TOP(KF) .EQ. 1.D0) THEN
               COEVAR(KF,KV) = -ZCOEF
               MINIMA(KF) = '(!)'
            ELSE
               COEVAR(KF,KV) = ZCOEF
               MINIMA(KF) = ' '
            END IF
            CALL LECV1(25,NAMEF,NAMEVF,TRF,ZCOEF,ENDFIC)
         END DO
      END DO
      CLOSE(25)
      CLOSE(32)
C Poids initials pour determiner la Fonction d'Utilite
      DO I=1,NOFOB
         POIDS(I) = 1. / NOFOB
      END DO
      RETURN
1000  FORMAT(A12,A1,F15.5)
2000  FORMAT(2A12,A1,F15.5)
      END
```

```
C=============================C
      SUBROUTINE COEFUT
C=============================C
C
C Parametres(NOCOLT,COEFOB,POIDS,COEVAR,NAMEF)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Calcul des coefficients de la fonction d'utilite
      DO J=1,NOCOLT
         COEFOB(J) = 0.D0
         DO I=1,NOFOR
            COEFOB(J) = COEFOB(J) + POIDS(I) * COEVAR(I,J)
         END DO
      END DO
C Assignation d'un nom a la fonction d'utilitee
      NAMEF = 'FUTILITEZW'
      RETURN
      END

C=============================C
      SUBROUTINE FUTZW
C=============================C
C
C Parametres(NOCOLT,NAMEF,NAMCOLT,COEFOB,INF)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Passage du fichier de variables de 32 a 42 pour
C l'optimisation(il faut mantenir le fichier 32)
      OPEN(32,ACCESS='SEQIN')
      OPEN(42,ACCESS='SEQOUT')
      ENDFIC = 'NON'
      CALL LECV1(32,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         WRITE(42,2000)NAMEL,NAMEVL,TRL,ZVAL
         CALL LECV1(32,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      END DO
      CLOSE(32)
C Apport des coefficients de la fonction d'utilite
C au fichier de variables
      TRF = 'N'
      DO J=1,NOCOLT
         WRITE(42,2000)NAMEF,NAMCOT(J),TRF,COEFOB(J)
      END DO
      CLOSE(42)
      CALL SORT('SORT/RECORD:40/KEY:13,12/KEY:1,12
     *           FOR42.DAT FOR42.DAT')
C Enregistrer les informations de la fonction d'utilite pour Lamps
      OPEN(33,ACCESS='SEQOUT')
      ZCOEF = 0.D0
      WRITE(33,1000)NAMEF,TRF,ZCOEF
      CLOSE(33)
      CALL OPPLS(FICHD,INF,N1,N2,41,42,0)
      IF (INF .GT. 0)WRITE(5,3000)
      RETURN
1000  FORMAT(A12,A1,F15.5)
2000  FORMAT(2A12,A1,F15.5)
3000  FORMAT(//1X,'Les contraintes du probleme sont mal conçues')
      END
```

```
C================================C
      SUBROUTINE VARHB
C================================C
C
C Parametres(SOLUT,NVHB,COUTR,NOFOB,NOCOLT,VALFOB,COEVAR,NOCOL,X,
C             NAMVEC,NAMCOL,NAMCOT,COEFOB,NVHB,NAMVHB,NAMEF,COUTR,INF)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Lecture des noms des variables de decision hors-base
      OPEN(24,ACCESS='SEQIN')
      KV = 0
      NVHB = 0
      ENDFIC = 'NON'
      CALL LECV2(24,NAMEF,NAMEVF,TRF,ZCOEF,CR,ENDFIC)
      CALL LECV2(24,NAMEF,NAMEVF,TRF,ZCOEF,CR,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         KV = KV + 1
         SOLUT(KV) = ZCOEF
         IF (ZCOEF .EQ. 0.D0) THEN
            NVHB = NVHB + 1
            NAMVHB(NVHB) = NAMEVF
            COUTR(NVHB) = CR
         END IF
         CALL LECV2(24,NAMEF,NAMEVF,TRF,ZCOEF,CR,ENDFIC)
      END DO
C Calcul des valeurs des f.o., pour la solution
C qui optimise la foction d'utilite
      DO I=1,NOFOB
         VALFOB(I) = 0
         DO J=1,NOCOLT
            VALFOB(I) = VALFOB(I) + COEVAR(I,J) * SOLUT(J)
         END DO
      END DO
C Valeurs des variables de decision
      IE = 1
      NOCOL = 0
      DO I = 1,NOCOLT
         IF (NAMCOT(I) .EQ. NAMVEC(IE)) THEN
            IE = IE +1
         ELSE
            NOCOL = NOCOL + 1
            X(NOCOL) = SOLUT(I)
            NAMCOL(NOCOL) = NAMCOT(I)
         END IF
      END DO
      CLOSE(24,DISPOSE='EXPUNGE')
```

```
C Optimisation de variables hors-base
      DO K=1,NOCOLT
         COFAUX(K) = COEFOB(K)
      END DO
      DO KF=1,NVHB
         J = 1
         DO WHILE(NAMCOT(J) .NE. NAMVHB(KF))
            J = J + 1
         END DO
         DO K=1,NOCOLT
            COEFOB(K) = COFAUX(K)
         END DO
         COEFOB(J) = COEFOB(J) + (COUTR(KF) + 0.01)
         NAMEF = NAMVHB(KF)
         CALL FUTZW
         IF (INF .GT. 0) GO TO 3000
      END DO
      CLOSE(24)
3000  RETURN
1000  FORMAT(A12,A1,F15.5)
2000  FORMAT(2A12,A1,F15.5)
      END
```

```
C================================C
      SUBROUTINE OPTVHB
C================================C
C
C Parametres(VALVAR,NVHB,NOCOLT,COEVAR,WVHB,VALFOB)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Lecture des valeurs optimales des v.h.b.
      OPEN(24,ACCESS='SEQIN')
      ENDFIC = 'NON'
      KF = 0
      CALL LECV1(24,NAMEF,NAMEVF,TRF,ZCOEF,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         NAMAUX = NAMEF
         KF = KF + 1
         KV = 1
         CALL LECV1(24,NAMEF,NAMEVF,TRF,ZCOEF,ENDFIC)
         DO WHILE((NAMAUX .EQ. NAMEF) .AND. (ENDFIC .EQ. 'NON'))
            VALVAR(KF,KV) = ZCOEF
            KV = KV + 1
            IF (NAMEVF .EQ. NAMVHB(KF)) THEN
               VOVHB(KF) = ZCOEF
            END IF
            CALL LECV1(24,NAMEF,NAMEVF,TRF,ZCOEF,ENDFIC)
         END DO
      END DO
      CLOSE(24,DISPOSE='EXPUNGE')
C Calcul des valeurs optimuns des f.o
C pour les variables hors-base
      DO J=1,NVHB
         DO I=1,NOFOB
            VFOVHB(I,J) = 0
            DO K=1,NOCOLT
               VFOVHB(I,J) = VFOVHB(I,J) + COEVAR(I,K)*VALVAR(J,K)
            END DO
         END DO
      END DO
C Calcul des taux de substitution pour les v.h.b.
      DO L1=1,NVHB
         J = L1
         DO L2=1,NOFOB
            I = L2
            WVHB(I,J) = (VALFOB(I) - VFOVHB(I,J)) / VOVHB(J)
         END DO
      END DO
      RETURN
      END
```

```
C==============================C
      SUBROUTINE SEVHB
C==============================C
C
C Parametres(NOFOB,WVHB,IND,NAMVHB,NVHB)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Elimination des v.h.b. qui ont des taux identiques
      DO I=1,NOFOB-1
         K = I + 1
         DO WHILE (K .LE. NOFOB)
            J = 1
            IND = 0
            DO WHILE((IND .EQ. 0) .AND. (J .LE. NVHB))
               IF (WVHB(I,J) .NE. WVHB(K,J)) IND = 1
               J = J + 1
            END DO
            IF (IND .EQ. 0) THEN
               NAMVHB(I) = ' '
               K = NVHB + 1
            ELSE
               K = K + 1
               IND = 0
            END IF
         END DO
      END DO
C Elimination des v.h.b. qui donnent des taux positifs uniquement
      DO I=1,NOFOB
         IF (NAMVHB(I) .NE. ' ') THEN
            IND = 0
            J = 1
            DO WHILE((IND .EQ. 0) .AND. (J .LE. NVHB))
               IF (WVHB(I,J) .LT. 0) IND = 1
               J = J + 1
            END DO
            IF (IND .EQ. 0) THEN
               NAMVHB(I) = ' '
            ELSE
               J = NVHB + 1
            END IF
         END IF
      END DO
      RETURN
      END
```

```
C==============================C
      SUBROUTINE VEFFIC
C==============================C
C
C Parametres(NVHB,RDTAUX,COUTR,NAMVHB,KCLEF)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Determination des variables hors-base efficaces
      KCLEF = 0
      DO L1=1,NVHB
         J = L1
         COUTR(J) = 0.D0
         IF (NAMVHB(J) .NE. ' ') THEN
C Appel bor minimiser la fonction correspondant
C aux taux d'une v.h.b. selectionnee
            CALL MINEFF(J)
         END IF
      END DO
      IF (KCLEF .EQ. 0) GO TO 1000
      DO I=1,100
         RDTAUX(I) = 0
      END DO
      DO L1=1,NVHB
         J = L1
         IF (COUTR(J) .GE. 0.D0) THEN
            NAMVHB(J) = ' '
         ELSE
            CALL DEMAND(J)
         END IF
      END DO
1000  RETURN
      END
```

```
C-----------------------------------C
      SUBROUTINE DEMAND(J)
C-----------------------------------C
C
C Parametres(NOFOB,WVHB,MINIMA,NAMFOB,VALFOB,TOP,RDTAUX)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Demande de prise de decision por les taux proposes
      WRITE(5,2000)
      DO L=1,NOFOB
         I = L
         ZVAL = - WVHB(I,J)
         WRITE(5,3000)MINIMA(I),NAMFOB(I),
     1                VIMP(VALFOB(I),I,TOP(I)),ZVAL
      END DO
      WRITE(5,3500)
      WRITE(5,4000)
      WRITE(5,5000)
10    READ(5,6000)IR
      IF (IR .NE. 1 .AND. IR .NE. 2 .AND. IR .NE. 3) THEN
         WRITE(5,7000)
         GO TO 10
      END IF
      RDTAUX(J) = IR
      RETURN
2000  FORMAT('1',6X,'Pour le suivant Taux de substitution',//4X,
     *'OBJECTIF',7X,'SOLUTION ACTUELLE',5X,'TAUX DE SUBSTITUTION',
     */4X,8('='),7X,17('='),5X,19('='))
3000  FORMAT(1X,A3,A12,3X,F15.5,7X,SP,F15.5)
3500  FORMAT(/1X,'(!)---> L''objectif dont son nom est precede',
     1 ' par ce signal est a MINIMISER')
4000  FORMAT(/10X,'1: Vous l''acceptez'/10X,'2: Vous ne l''acceptez',
     * ' pas',/10X,'3: Vous etes indifferent')
5000  FORMAT(//15X,'Votre choix:',$)
6000  FORMAT(I1)
7000  FORMAT(/1X,'Votre reponse doit etre 1, 2, ou 3:',$)
      END
```

```
C--------------------------------C
      SUBROUTINE MINEFF(J)
C--------------------------------C
C
C Parametres(LZW,WVHB,NAMVHB,NL,NVHB,KCLEF,INF)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Minimisation des fonctions objectifs generees pour
C les taux des v.h.b. selectionnes
C Fonction objectif pour resoudre les p.l.
C que nous permetra determiner les v.h.b efficaces
      OPEN(33,ACCESS='SEQOUT')
      TRF = 'N'
      ZCOEF = 1.D0
      NAMEF = 'FO'
      WRITE(33,2000)NAMEF,TRF,ZCOEF
      CLOSE(33)
      OPEN(31,ACCESS='SEQOUT')
      OPEN(42,ACCESS='SEQOUT')
      DO I=1,NOFOB
         WRITE(42,3000)NAMEF,LZW(I),TRF,WVHB(I,J)
      END DO
      ZVAL = 0.D0
      TRL = 'G'
      K=1
      DO WHILE(K .LE. NVHB)
         IF ((K .NE. J) .AND. (NAMVHB(K) .NE. ' ')) THEN
            WRITE(31,2000)NL(K),TRL,ZVAL
            DO I=1,NOFOB
               WRITE(42,3000)NL(K),LZW(I),TRL,WVHB(I,K)
            END DO
         END IF
         K = K + 1
      END DO
      TRL = 'E'
      WRITE(31,2000)NL(NVHB + 1),TRL,ZCOEF
      DO I=1,NVHB
         WRITE(42,3000)NL(NVHB + 1),LZW(I),TRL,ZCOEF
      END DO
      CLOSE(31)
      CLOSE(42)
      CALL SORT('SORT/RECORD:40/KEY:13,12/KEY:1,12
     *           FOR42.DAT FOR42.DAT')
      CALL OPPLS(FICHD,INF,N1,N2,31,42,0)
      IF (INF .GT. 0) GO TO 1000
      OPEN(24,ACCESS='SEQIN')
      ENDFIC='NON'
      CALL LECV1(24,NAMEF,NAMEVF,TRF,ZCOEF,ENDFIC)
      COUTR(J) = ZCOEF
      IF (ZCOEF .LT. 0.D0) THEN
         KCLEF = KCLEF + 1
      END IF
1000  CLOSE(24,DISPOSE='EXPUNGE')
C A remetre close(24,dispose='expunge')
      RETURN
2000  FORMAT(A12,A1,F15.5)
3000  FORMAT(2A12,A1,F15.5)
      END
```

```
C================================C
      SUBROUTINE FVEF
C================================C
C
C Parametres(NVHB,PDTAUX,KCLEFA)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Generation de la matrice de contraintes avec les
C taux correspondant aux variables efficaces
      IND = 0
C Tester s'il y a des reponses d'acceptation au taux propose
      DO J=1,NVHB
         IF (RDTAUX(J) .EQ. 1) IND = 1
      END DO
      IF (IND .EQ. 0) GO TO 1000
C Generation des fichiers avec les taux
      OPEN(36,ACCESS='APPEND')
      OPEN(37,ACCESS='APPEND')
      DO L1=1,NVHB
         J = L1
         IF (RDTAUX(J) .EQ. 1) THEN
            KCLEFA = KCLEFA + 1
            TRL = 'L'
            ZVAL = -EPSI
            CALL AJOULV(J)
         ELSE IF (PDTAUX(J) .EQ. 2) THEN
            KCLEFA = KCLEFA + 1
            TRL = 'G'
            ZVAL = EPSI
            CALL AJOULV(J)
         END IF
      END DO
      CLOSE(36)
      CLOSE(37)
1000  RETURN
      END


C--------------------------------C
      SUBROUTINE AJOULV(J)
C--------------------------------C
C
C Parametres(KCLEFA,NL,LZW,WVHB)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Ajoute de contraintes
      IF (KCLEFA .EQ. 1) THEN
         CALL PFOIS(J)
      END IF
      WRITE(36,1000)NL(KCLEFA),TRL,ZVAL
      DO I=1,NOFOB
         WRITE(37,2000)NL(KCLEFA),LZW(I),TRL,WVHB(I,J)
      END DO
      RETURN
1000  FORMAT(A12,A1,F15.5)
2000  FORMAT(2A12,A1,F15.5)
      END
```

```
C-----------------------------------C
      SUBROUTINE PFOIS(J)
C-----------------------------------C
C
C Parametres(NOFOB,LZW,WVHB)
C
      INCLUDE 'COMPL.MOB/NOLIST'
C
C Stockage des coefficients de taux, correspondant
C a la premiere v. efficace, comme fonction objectif,
C et c'est la meme pour toutes les iterations
      TRF = 'M'
      ZCOEF = 0.D0
      NAMEF = 'FOLAMBDA'
      OPEN(38,ACCESS='SEQOUT')
      WRITE(38,1000)NAMEF,TRF,ZCOEF
      CLOSE(38)
      DO I=1,NOFOB
         WRITE(37,2000)NAMEF,LZW(I),TRF,WVHB(I,J)
      END DO
      TRF = 'E'
      ZCOEF = 1.D0
      NAMEL = 'UNITE'
      WRITE(36,1000)NAMEL,TRF,ZCOEF
      DO I=1,NOFOB
         WRITE(37,2000)NAMEL,LZW(I),TRF,ZCOEF
      END DO
      RETURN
1000  FORMAT(A12,A1,F15.5)
2000  FORMAT(2A12,A1,F15.5)
      END


C==================================C
      SUBROUTINE PETRAV
C==================================C
C
C Parametres(INF)
C
      INCLUDE 'COMPL.MOB/NOLIST'

      OPEN(38,ACCESS='SEQIN')
      OPEN(33,ACCESS='SEQOUT')
      CALL LECL1(38,NAMEF,TRF,ZCOEF,ENDFIC)
      WRITE(33,1000)NAMEF,TRF,ZCOEF
      CLOSE(38)
      CLOSE(33)
      CALL SORT('SORT/RECORD:40/KEY:13,12/KEY:1,12
     *           FOR37.DAT FOR42.DAT')
      CALL CPPLS(FICHD,INF,N1,N2,36,42,0)
      IF (INF .GT. 0) WRITE(5,3000)
      RETURN
1000  FORMAT(A12,A1,F15.5)
2000  FORMAT(2A12,A1,F15.5)
3000  FORMAT(//1X,'Les decisions sur les demandes de choix de'
     *'variations des f.o. sont incoherents')
      END
```

```
C================================C
      SUBROUTINE NPOIDS
C================================C
C
C Parametres(POIDS9
C
      INCLUDE 'COMPL.MOB/NOLIST'

      OPEN(24,ACCESS='SEQIN')
      K = 0
      ENDFIC = 'NON'
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      DO WHILE(ENDFIC .EQ. 'NON')
         K = K + 1
         POIDS(K) = ZVAL
         CALL LECV1(24,NAMEL,NAMEVL,TRL,ZVAL,ENDFIC)
      END DO
      CLOSE(24,DISPOSE='EXPUNGE')
      RETURN
      END
```

```
C*************************************************$
      SUBROUTINE MAJ
C*************************************************$

      INCLUDE 'COMPL.MOB/NOLIST'

      WRITE(5,2000)
C Selection du type d'atualisation a effectuer
10    READ(5,3000)R
      IF (R .EQ. '1') THEN
C Acualisation des RHS
         CALL MAJTI
      ELSE IF(R .EQ. '2') THEN
C Actualisation des termes de la matrice
         CALL MAJCOE
      ELSE
         WRITE(5,4000)
         GO TO 10
      END IF
      RETURN
2000  FORMAT('1',20X,'ACTUALISATIONS'/21X,14('=')//1X,'Tapez',
     1 ' 1: si vous voulez changer la valeur d''un terme inde',
     2 'pendant'//1X,'Tapez 2: si vous voulez changer la valeur',
     3 ' d''un coefficient'//10X,'Votre choix:',$)
3000  FORMAT(A1)
4000  FORMAT(1X,'Vous devez taper 1 ou 2 :',$)
      END
```

```
C================================C
      SUBROUTINE MAJTI
C================================C

      INCLUDE 'COMPL.MOB/NOLIST'
C
C Generation du fichier qui contient les changements a realiser
C sur les RHS (introduites de facon interactive)
      WRITE(5,1000)
      OPEN(26,FILE='MAJ.TIN',ACCESS='SEQOUT')
      R = 'Y'
      DO WHILE (R .EQ. 'Y' .OR. R .EQ. 'y')
         WRITE(5,2000)
         READ(5,3000)NAMEL
         WRITE(5,4000)
         READ(5,*)ZVAL
         WRITE(26,6000)NAMEL,ZVAL
         WRITE(5,8000)
         CALL TESTR(R)
      END DO
      CLOSE(26)
C Triement du fichier de mouvements par nom de ligne
      CALL SORT('SORT/RECORD:28/KEY:1,12 MAJ.TIN MAJ.TIN')
C Appel a la procedure qui effectue l'actualisation
      CALL FUSITI
      RETURN
1000  FORMAT('1',10X,'ACTUALISATION D''UN TERME INDEPENDANT'/,
     1 11X,36('='))
2000  FORMAT(/1X,'Tapez le nom de l''equation:',$)
3000  FORMAT(A12)
4000  FORMAT(/1X,'Tapez la valeur du terme independant:',$)
6000  FORMAT(A12,1X,F15.5)
8000  FORMAT(/1X,'Voulez-vous changer la valeur d''un autre terme',
     1 ' independant(Y ou N):',$)
      END
```

```
C============================C
      SUBROUTINE FUSITI
C============================C

      INCLUDE 'COMPL.MOB/NOLIST'

      CHARACTER*3 FIN1,FIN2
C
C Actualisations des RHS en utilisant la technique de fusion de fichiers
      OPEN(26,FILE='MAJ.TIN',ACCESS='SEQIN')
      OPEN(27,FILE='FICH.LIG',ACCESS='SEQOUT')
      OPEN(21,FILE=FICHL,ACCESS='SEQIN')
      FIN1 = 'NON'
      FIN2 = 'NON'
      CALL LECL1(21,NAMEF,TRF,ZCOEF,FIN1)
      CALL LECL1(26,NAMEL,TRL,ZVAL,FIN2)
      DO WHILE(FIN1 .EQ. 'NON' .AND. FIN2 .EQ. 'NON')
         IF (NAMEF .LT. NAMEL) THEN
            WRITE(27,2000)NAMEF,TRF,ZCOEF
            CALL LECL1(21,NAMEF,TRF,ZCOEF,FIN1)
         ELSE IF (NAMEF .EQ. NAMEL) THEN
            WRITE(27,2000)NAMEF,TRF,ZVAL
            CALL LECL1(21,NAMEF,TRF,ZCOEF,FIN1)
            CALL LECL1(26,NAMEL,TRL,ZVAL,FIN2)
         ELSE
            WRITE(5,3000)NAMEL
            CALL LECL1(26,NAMEL,TRL,ZVAL,FIN2)
         END IF
      END DO
      IF (FIN1 .EQ. 'NON') THEN
         DO WHILE(FIN1 .EQ. 'NON')
            WRITE(27,2000)NAMEF,TRF,ZCOEF
            CALL LECL1(21,NAMEF,TRF,ZCOEF,FIN1)
         END DO
      ELSE IF (FIN2 .EQ. 'NON') THEN
         DO WHILE(FIN2 .EQ. 'NON')
            WRITE(5,3000)NAMEL
            CALL LECL1(26,NAMEL,TRL,ZVAL,FIN2)
         END DO
      END IF
      CLOSE(21,DISPOSE='EXPUNGE')
      CLOSE(26,DISPOSE='EXPUNGE')
      CLOSE(27,DISPOSE='RENAME',FILE=FICHL)
      RETURN
2000  FORMAT(A12,A1,F15.5)
3000  FORMAT(/1X,'Il n''existe pas l''equation avec le nom: ',A12)
      END
```

```
C================================C
      SUBROUTINE MAJCOE
C================================C

      INCLUDE 'COMPL.MOB/NOLIST'
C
C Generation du fichier qui contient les changements des
C termes de la matrice (introduites de facon interactive)
      WRITE(5,1000)
      OPEN(26,FILE='MAJ.COE',ACCESS='SEQOUT')
      R = 'Y'
      DO WHILE ( R .EQ. 'Y' .OR. R .EQ. 'y')
         WRITE(5,4000)
         READ(5,3000)NAMEL
         WRITE(5,2000)
         READ(5,3000)NAMEVL
         WRITE(5,5000)
         READ(5,*)ZVAL
         WRITE(26,6000)NAMEL,NAMEVL,ZVAL
         WRITE(5,9000)
         CALL TESTR(R)
      END DO
      CLOSE(26)
C Triement du fichier de mouvements par nom de colonne
C et nom de ligne
      CALL SORT('SORT/RECORD:40/KEY:13,12/KEY:1,12 MAJ.COE MAJ.COE')
C Appel a la procedure qui effectue l'actualisation
      CALL FUSICO(ISW)
C Actualisation au fichier des termes des objectifs (s'il y en a)
      IF (ISW .EQ. 0) THEN
         CLOSE(28,DISPOSE='EXPUNGE')
      ELSE
         CALL MAJOBJ
      END IF
      RETURN
1000  FORMAT('1',10X,'ACTUALISATION D''UN COEFFICIENT'/11X,30('='))
2000  FORMAT(/1X,'Tapez le nom de la variable:',$)
3000  FORMAT(A12)
4000  FORMAT(/1X,'Tapez le nom de l''equation:',$)
5000  FORMAT(/1X,'Tapez la valeur du coefficient:',$)
6000  FORMAT(2A12,1X,F15.5)
9000  FORMAT(/1X,'Voulez-vous changer la valeur d''un autre',
     1 ' coefficient?(Y ou N):',$)
      END
```

```
C================================C
      SUBROUTINE FUSICO(ISW)
C================================C

      INCLUDE 'COMPL.MOB/NOLIST'

      CHARACTER*3 FIN1,FIN2
C
C Actualisation des termes de la matrice en utilisant
C la technique de fusion de fichiers
      OPEN(26,FILE='MAJ.COE',ACCESS='SEQIN')
      OPEN(27,FILE='FICH.COE',ACCESS='SEQOUT')
      OPEN(28,FILE='FICH.OBJ',ACCESS='SEQOUT')
      OPEN(22,FILE=FICHV,ACCESS='SEQIN')
      FIN1 = 'NON'
      FIN2 = 'NON'
      ISW = 0
      CALL LECV1(22,NAMEF,NAMEVF,TRF,ZCOEF,FIN1)
      CALL LECV1(26,NAMEL,NAMEVL,TRL,ZVAL,FIN2)
      DO WHILE(FIN1 .EQ. 'NON' .AND. FIN2 .EQ. 'NON')
         IF (NAMEVF .LT. NAMEVL) THEN
            WRITE(27,2000)NAMEF,NAMEVF,TRF,ZCOEF
            CALL LECV1(22,NAMEF,NAMEVF,TRF,ZCOEF,FIN1)
         ELSE IF (NAMEVF .EQ. NAMEVL) THEN
            CALL SUITEA(ISW,FIN1,FIN2)
         ELSE
            CALL MESAGE
            CALL LECV1(26,NAMEL,NAMEVL,TRL,ZVAL,FIN2)
         END IF
      END DO
      IF (FIN1 .EQ. 'NON') THEN
         DO WHILE(FIN1 .EQ. 'NON')
            WRITE(27,2000)NAMEF,NAMEVF,TRF,ZCOEF
            CALL LECV1(22,NAMEF,NAMEVF,TRF,ZCOEF,FIN1)
         END DO
      ELSE IF (FIN2 .EQ. 'NON') THEN
         DO WHILE(FIN2 .EQ. 'NON')
            CALL MESAGE
            CALL LECV1(26,NAMEL,NAMEVL,TRL,ZVAL,FIN2)
         END DO
      END IF
      CLOSE(22,DISPOSE='EXPUNGE')
      CLOSE(26,DISPOSE='EXPUNGE')
      CLOSE(27,DISPOSE='RENAME',FILE=FICHV)
      RETURN
2000  FORMAT(2A12,A1,F15.5)
      END

C------------------------------C
      SUBROUTINE MESAGE
C------------------------------C

      INCLUDE 'COMPL.MOB/NOLIST'

      WRITE(5,3000)NAMEL,NAMEVL
      RETURN
3000  FORMAT(/1X,'Il n''existe pas un terme correspondant a',
     1 ' l''equation:',A12/1X,35X,'et a la variable:',A12)
      END
```

```
C=======================================C
      SUBROUTINE SUITEA(ISW,FIN1,FIN2)
C=======================================C

      INCLUDE 'COMPL.MOB/NOLIST'

      CHARACTER*3 FIN1,FIN2

      IF (NAMEF .LT. NAMEL) THEN
         WRITE(27,2000)NAMEF,NAMEVF,TRF,ZCOEF
         CALL LECV1(22,NAMEF,NAMEVF,TRF,ZCOEF,FIN1)
      ELSE IF (NAMEF .EQ. NAMEL) THEN
         WRITE(27,2000)NAMEF,NAMEVF,TRF,ZVAL
         IF (TRF .EQ. 'N') THEN
            ISW = 1
            WRITE(28,2000)NAMEF,NAMEVF,TRF,ZVAL
         END IF
         CALL LECV1(22,NAMEF,NAMEVF,TRF,ZCOEF,FIN1)
         CALL LECV1(26,NAMEL,NAMEVL,TRL,ZVAL,FIN2)
      ELSE
         CALL MESAGE
         CALL LECV1(26,NAMEL,NAMEVL,TRL,ZVAL,FIN2)
      END IF
1000  RETURN
2000  FORMAT(2A12,A1,F15.5)
      END
```

```
C================================C
      SUBROUTINE MAJOBJ
C================================C

      INCLUDE 'COMPL.MOB/NOLIST'

      CHARACTER*3 FIN1,FIN2
C Actualisation des termes des fonctions objectifs
      CLOSE(28)
      CALL SORT('SORT/RECORD:40/KEY:1,12/KEY:13,12 FICH.OBJ FICH.OBJ')
      OPEN(25,FILE=FICHO,ACCESS='SEQIN')
      OPEN(26,FILE='FICH.OBJ',ACCESS='SEQIN')
      OPEN(27,FILE='OBJ.MAJ',ACCESS='SEQOUT')
      FIN1 = 'NON'
      FIN2 = 'NON'
      CALL LECV1(25,NAMEF,NAMEVF,TRF,ZCOEF,FIN1)
      CALL LECV1(26,NAMEL,NAMEVL,TRL,ZVAL,FIN2)
      DO WHILE (FIN1 .EQ. 'NON' .AND. FIN2 .EQ. 'NON')
         DO WHILE(NAMEVF .LT. NAMEVL .OR. NAMEF .LT. NAMEL)
            WRITE(27,1000)NAMEF,NAMEVF,TRF,ZCOEF
            CALL LECV1(25,NAMEF,NAMEVF,TRF,ZCOEF,FIN1)
         END DO
            WRITE(27,1000)NAMEF,NAMEVF,TRF,ZVAL
            CALL LECV1(25,NAMEF,NAMEVF,TRF,ZCOEF,FIN1)
            CALL LECV1(26,NAMEL,NAMEVL,TRL,ZVAL,FIN2)
      END DO
      IF (FIN1 .EQ. 'NON') THEN
         DO WHILE(FIN1 .EQ. 'NON')
            WRITE(27,1000)NAMEF,NAMEVF,TRF,ZCOEF
            CALL LECV1(25,NAMEF,NAMEVF,TRF,ZCOEF,FIN1)
         END DO
      END IF
      CLOSE(25,DISPOSE='EXPUNGE')
      CLOSE(26,DISPOSE='EXPUNGE')
      CLOSE(27,DISPOSE='RENAME',FILE=FICHO)
      RETURN
1000  FORMAT(2A12,A1,F15.5)
      END

C********************************C
      SUBROUTINE FINIR
C********************************C

      OPEN(31,ACCESS='SEQOUT')
      OPEN(32,ACCESS='SEQOUT')
      OPEN(33,ACCESS='SEQOUT')
      OPEN(35,ACCESS='SEQOUT')
      OPEN(41,ACCESS='SEQOUT')
      OPEN(42,ACCESS='SEQOUT')
      OPEN(20,FILE='MATRIX.NAM')
      CLOSE(31,DISPOSE='EXPUNGE')
      CLOSE(32,DISPOSE='EXPUNGE')
      CLOSE(33,DISPOSE='EXPUNGE')
      CLOSE(35,DISPOSE='EXPUNGE')
      CLOSE(41,DISPOSE='EXPUNGE')
      CLOSE(42,DISPOSE='EXPUNGE')
      CLOSE(20,DISPOSE='EXPUNGE')
      RETURN
      END
```

I N T E R F A C E A V E C L A M P S

```
C*************************************************************S
      SUBROUTINE OPPLS(NOMFM,INF,NOLIG,NOCOL,NFL,NFV,IFCART)
C*************************************************************S

      INCLUDE 'LAMPS:ALXCOM.JHF/NOLIST'
      INCLUDE 'LAMPS:STXCOM.JHF/NOLIST'
      INCLUDE 'COMLA.MPS/NOLIST'
C
C                  DEFINITION DE VARIABLES
C                  =======================
C NOMFM: Nom du fichier qui contient le nom de la matrice de donnees
C MMNAME: Nom de la matrice de donnees
C MDIRNM: Nom de la directorie(exige par Lamps), ici c'est idem au
C         nom de la matrice de donnees(MMNAME)
C NLIGNE: Nom d'une equation
C NFOOBJ: Nom d'une foction objectif
C INF: Indique si le probleme a une solution optimale(valeur 0)
C      ou pas(valeur 1)
C NOLIG: Nombre de lignes de la matrice de donnees
C NOCOL: Nombre de colonnes de la matrice de donnees
C NFL: Numero du fichier qui contient l'information sur les noms
C      des equations et valeurs des termes independants
C NFV: Numero du fichier qui contient l'information sur les termes
C      de la matrice de donnees
C
C                      ======================
```

```
C=====================================================================C
C  SPECIFICATIONS DU TYPE DES VARIABLES GLOBALES DU PROGRAMME
C  OPPLS ET DECLARATIONS COMMON AVEC ETIQUETTE
C=====================================================================C
      CHARACTER*10 NOMFM
      COMMON /NUM/ NDIRNM(2),MNNAME(2)
      COMMON /ABC/ NLIGNF(3),ZVAL,NFOOBJ(3),IT,ENDFIC
```

```
C
C Optimisation d'un problemme de programmation lineaire
C en utilisant subroutines du logiciel Lamps
C
      NCOLFT=500
      IFVIRT=0
C Les noms des fonctions objectifs doivent etre stockees
C sur le fichier 33
      OPEN(UNIT=33,ACCESS='SEQIN')
      ENDFIC='NON'
      NOLIG=0
      K=0
      CALL LECL(33)
      DO WHILE(ENDFIC.EQ.'NON')
         DO I=1,3
            NFOOBJ(I)=NLIGNE(I)
         END DO
         MAXMIN=ZVAL
C Lecture de la matrice du probleme
         CALL GM(K,NFL,NFV,NOMFM)
C Fermeture de fichiers lamps
         CALL FERMER
C Optimisation du probleme
         CALL PRIMEX(K,MAXMIN)
C Fermeture de fichiers lamps
         CALL FERMER
C Lecture des resultats
         CALL RR(INF,NOCOL,IECART)
C Test si le probleme a solution optimale
         IF (INF .GT. 1) GO TO 1000
C Fermeture de fichiers lamps
         CALL FERMER
C Lecture du nom de la suivante fonction objective
         NOLIG=NOLIG + 1
         K=1
         ENDFIC='NON'
         CALL LECL(33)
      END DO
1000  CLOSE(33)
      RETURN
      END
```

```
C=======================================C
      SUBROUTINE GM(K,NFL,NFV,NOMFM)
C=======================================C

      INCLUDE 'LAMPS:ALXCOM.JHF'
      INCLUDE 'COMLA.MPS/NOLIST'
C
C Generation de la matrice dans le fichier de travail
C lamps, en utilisant la subroutine Matrix Generator
C
C Appel lecture du nom de la matrice
      IF (K.EQ.0) THEN
         CALL LECNMA(NOMFM)
      END IF
C Appel subroutine lamps pour la creation de fichiers
C demandes par lamps avec le nom de la matrice du probleme
      CALL NOMMA
C Subroutine lamps pour la lecture des noms de lignes
      CALL LIGNES(NFL)
C Subroutine lamps pour la lecture des noms et coefficients
C de variables
      CALL COLONS(NFV)
C Subroutine lamps pour lecture du nom et coefficients
C de termes independants
      CALL TI(NFL)
C Fin d'utilisation de la subroutine lamps pour lecture de donnes
      CALL MGEND(NCOL,NRHS,NRNG,NNZRO)
      RETURN
      END

C-----------------------------------C
      SUBROUTINE LECNMA(NOMFM)
C-----------------------------------C

      INCLUDE 'LAMPS:ALXCOM.JHF/NOLIST'
      INCLUDE 'COMLA.MPS/NOLIST'
C
C Lecture du nom de la matrice du probleme
C assigne par l'utilisteur
C
      OPEN(UNIT=20,FILE=NOMFM,MODE='ASCII',
     *     ACCESS='SEQIN',READONLY,STATUS='OLD')
      READ(20,1000)(MMNAME(I),I=1,2)
      DO I=1,2
         MDIRNM(I) = MMNAME(I)
      END DO
      CLOSE(20)
      RETURN
1000  FORMAT(2A4)
      END
```

```
C--------------------------C
      SUBROUTINE NOMMA
C--------------------------C

      INCLUDE 'LAMPS:ALXCOM.JHF/NOLIST'
      INCLUDE 'COMLA.MPS/NOLIST'
      INTEGER*4 MDESC(6)
C
C Creation des fichiers necessaires pour lamps
C et qu'auront le meme nom de la matrice
C
      CALL MGNAME(MDIRNM,MMNAME,'PROBLEME  MULTI-OBJECTIVE')
      RETURN
      END

C----------------------------------C
      SUBROUTINE LIGNES(NFL)
C----------------------------------C

      INCLUDE 'LAMPS:ALXCOM.JHF/NOLIST'
      INCLUDE 'COMLA.MPS/NOLIST'
C
C Passage des informations sur les lignes au
C fichier de travail lamps
C
      OPEN(UNIT=NFL,ACCESS='SEQIN')
      CALL MGROW(IT,NLIGNE)
      ENDFIC='NON'
      CALL LECL(NFL)
      DO WHILE (ENDFIC .NE. 'OUI')
            CALL MGROW(IT,NLIGNE)
            CALL LECL(NFL)
      END DO
      CALL MGROWE(NROW,NBOUND)
      CLOSE(NFL)
      RETURN
      END

C----------------------------------C
      SUBROUTINE LECL(NUMERO)
C----------------------------------C

      INCLUDE 'LAMPS:ALXCOM.JHF/NOLIST'
      INCLUDE 'COMLA.MPS/NOLIST'
C Lecture de l'information correspondant a une ligne
C de la matrice
      READ(NUMERO,1000,END=2000)NLIGNE,IT,ZVAL
      RETURN
2000  ENDFIC='OUI'
      RETURN
1000  FORMAT(3A4,A1,F15.5)
      END
```

```
C-----------------------------------C
      SUBROUTINE COLONS(NFV)
C-----------------------------------C

      INCLUDE 'LAMPS:ALXCOM.JHF/NOLIST'
      INCLUDE 'COMLA.MPS/NOLIST'
      INTEGER*4 MCOLNM(3),COLAUX(3)
C
C Passage des informations sur les variables au
C fichier de travail lamps
C
      OPEN(UNIT=NFV,ACCESS='SEQIN')
      ENDFIC='NON'
      CALL LECV(MCOLNM,NFV)
      DO WHILE (ENDFIC .NE. 'OUI')
         DO I=1,3
            COLAUX(I)=MCOLNM(I)
         END DO
         CALL MGCOL(MCOLNM)
         DO WHILE ((IND(MCOLNM,COLAUX).EQ.0).AND.(ENDFIC.EQ.'NON'))
            IF (IT.EQ.'N') THEN
               IF (IND(NLIGNE,NFOBJ).EQ.0) THEN
                  CALL MGEL(0,NLIGNE,ZVAL)
               END IF
            ELSE
                CALL MGEL(0,NLIGNE,ZVAL)
            END IF
            CALL LECV(MCOLNM,NFV)
         END DO
      END DO
      CLOSE(NFV)
      RETURN
      END


C---------------------------------------C
      FUNCTION IND(MCOLNM,COLAUX)
C---------------------------------------C

      INTEGER*4 MCOLNM(3),COLAUX(3)
      K=0
      DO I=1,3
         IF (MCOLNM(I) .NE. COLAUX(I)) THEN
            K=1
         END IF
      END DO
      IF (K .EQ. 0) THEN
         IND=0
      ELSE
         IND=1
      END IF
      RETURN
      END
```

```
C--------------------------------------C
      SUBROUTINE LECV(MCOLNM,NFV)
C--------------------------------------C

      INCLUDE 'LAMPS:ALXCOM.JHF/NOLIST'
      INCLUDE 'COMLA.MPS/NOLIST'
      INTEGER*4 MCOLNM(3),COLAUX(3)
C Lecture des informations correspondants aux
C variables de la matrice
      READ(NFV,1000,END=2000)NLIGNE,MCOLNM,IT,ZVAL
      RETURN
2000  ENDFIC='OUI'
      RETURN
1000  FORMAT(3A4,3A4,A1,F15.5)
      END

C============================C
      SUBROUTINE TI(NFL)
C============================C

      INCLUDE 'LAMPS:ALXCOM.JHF/NOLIST'
      INCLUDE 'COMLA.MPS/NOLIST'
      REAL*8 ZVAL
C
C Passage des informations sur les T.I.
C au fichier de travail lamps
C
      OPEN(UNIT=NFL,ACCESS='SEQIN')
      ENDFIC='NON'
      CALL MGKHS ('RHS1       ')
      CALL LECL(NFL)
      DO WHILE (ENDFIC .NE. 'OUI')
         IF (IT.NE.'N') THEN
             CALL MGFL(0,NLIGNE,ZVAL)
         END IF
         CALL LECL(NFL)
      END DO
      CLOSE(NFL)
      RETURN
      END
```

```
C=======================================C
      SUBROUTINE PRIMEX(K,MAXMIN)
C=======================================C

      INCLUDE 'LAMPS:ALXCOM.JHF/NOLIST'
      INCLUDE 'COMLA.MPS/NOLIST'
C
C Optimisation du probleme
C
      DO I=1,2
         MINFIL(I) = -1
         MATRXO(I) = MMNAME(I)
         MNAME2(I) = -1
      END DO
      IFRCOM = 0
      ISOLNM =-1
      IFBTCH=1
      IMXITM = 999999
      IKEY=1
      IFMINI = MAXMIN
C Appel a la subroutine lamps (PRIMAL) pour la
C resolution du probleme
      CALL PRIMAL
      RETURN
      END

C============================C
      SUBROUTINE FERMER
C============================C

      INCLUDE 'LAMPS:ALXCOM.JHF/NOLIST'
C
C Fermeture des fichiers geres par lamps, c'est necessaire
C apres chaque fin d'execution d'une subroutine lamps
C
      CALL CLOSEF(ISOLF1,KSTAT)
      CALL CLOSEF(IRUNIT,KSTAT)
      CALL CLOSEF(ISOLV,KSTAT)
      RETURN
      END
```

```
C=============================================C
      SUBROUTINE RR(INF,NOCOL,IECART)
C=============================================C

      INCLUDE 'LAMPS:PRECIS.JHF/NOLIST'
      INCLUDE 'LAMPS:RWCOM.JHF'
      INCLUDE 'COMLA.MPS/NOLIST'
C
C Lecture des resultats de l'optimisation
C utilisant la subroutine Report Writer
      OPEN(24,ACCESS='APPEND',CARRIAGECONTROL='LIST')
      CALL RWOPEN(MMNAME,'SOL',NSOL,NROW,NCOL)
      NOCOL=NCOL
      CALL RWFIND(NSOL)
      IF ((NINF .EQ. 0) .AND. ISTATS .EQ. 1) THEN
         INF = 0
      ELSE
         INF = 1
      END IF
      IF (NINF .GT. 0) GO TO 3000
      WRITE(24,1000)MOBJ,MOBJ,DOBJVL
C Lecture des valeurs des variables d'ecart(optionelle)
      IF (IECART .EQ. 1) THEN
         OPEN(26,ACCESS='APPEND',CARRIAGECONTROL='LIST')
         DO IROW=1,NROW
            L=IROW
            CALL RWROW(L)
            WRITE(26,1000)MNAME,MNAME,DPI,DVALUE
         END DO
         CLOSE(26)
      END IF
C Lecture de la solution
      DO ICOL=1,NCOL
         L=ICOL
         CALL RWCOL(L)
         WRITE(24,1000)MOBJ,MNAME,DVALUE,DPI
      END DO
3000  CLOSE(24)
      RETURN
1000  FORMAT(3A4,3A4,1X,3F15.5)
2000  FORMAT(I3)
      END
```